UNIVERSITY OF ALBERTA

Two-Handed 3D User Interfaces

BY

Christopher David Shaw

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

DEPARTMENT OF COMPUTING SCIENCE

Edmonton, Alberta
Spring 1997

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced with the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-21632-2

Canada

UNIVERSITY OF ALBERTA

LIBRARY RELEASE FORM

NAME OF AUTHOR: Christopher David Shaw

TITLE OF THESIS: Two-Handed 3D User Interfaces

DEGREE: Doctor of Philosophy

YEAR THIS DEGREE GRANTED: 1997

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

(Signed) .....

Christopher David Shaw
6 Alta Drive
Bristol Pond
RR #3, Stouffville,
Ontario
L4A 7X4
cdshaw@acm.org

Date: Jan 31 '97

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Two-Handed 3D User Interfaces** submitted by Christopher David Shaw in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dr. Mark W. Green (Supervisor)

for Dr. Frederick P. Brooks, Jr. (External)

Dr. Roger W. Toogood (Examiner)

Dr. Stan Cabay (Examiner)

Dr. John W. Buchanan (Examiner)

Dr. Janelle Harms (Examiner)

Date: 31. Jan. 97

To Diana

Abstract

Computer-aided geometric modeling of free-form surfaces is often a time-consuming task. Part of the difficulty arises because the traditional 2D input and output devices in use today are not geometrically matched with the task. One solution to this problem is to use devices that yield 3D information to perform 3D tasks.

This thesis presents the design and implementation of a two-handed 3D design system called THRED (Two Handed Refining EDitor), where the user sits at a desk and manipulates two 6 degree-of-freedom (DOF) trackers with three buttons each. The interaction techniques used to make this style work are presented, along with the integration of these techniques into a complete system.

Next, this thesis reports an user study in which subjects used THRED and two other interfaces having THRED's geometry operations to perform multiple timed trials of two simple geometric editing tasks. The other experimental interfaces were the Mouse-Based interface (a simplified clone of the Alias modeler), and the One-Handed interface (a version of THRED that uses one button-enhanced 6 DOF tracker in the right hand and presses keyboard keys with the left hand). Subjects also responded to a questionnaire asking about preferences, and inquiring about the subject's state of comfort over the course of the experiment.

The results of the experiment show a strong general preference for THRED over the other two interfaces, and preference for all of THRED's interface details over the One-Handed interface. Only Mouse-Based menu operations and vertex picking were preferred over THRED. Trial times indicate better performance in THRED over the Mouse-Based interface in the simple 2D task, and better performance than the One-Handed interface on the 3D task. The pain and fatigue survey showed THRED and the Mouse-Based interface showed no statistically significant increase in discomfort throughout the trials, while the One-Handed interface showed increase in left arm fatigue. Finally,

timings indicate that after THRED users had received only 16 minutes more training and practice than Mouse-Based users, THRED users were able to outperform Mouse-Based users on the first timed trial of the entire experiment.

Acknowledgements

Jiandong (JD) Liang influenced me greatly while I was in the Graphics Lab at Alberta, and many aspects of this work would not have been possible without him.

Thanks to Mark Green, John Buchanan, Paul Ferry, Oleg Veryovka, Lloyd White, and Hongzhi Wang.

Many thanks to Fred Brooks, Jr. for reading this work on short notice and in a big hurry.

Thanks to Diana Bang Gromala for everything else!

Contents

1 Introduction 1

1.1 The New Interface . . . 2
1.2 Evaluation of the Interface . . . 2
1.3 The Thesis . . . 3
1.4 Contributions – User Study . . . 4
1.4.1 Learning . . . 5
1.4.2 Times . . . 5
1.4.3 Pain and Fatigue . . . 5
1.4.4 Preferences . . . 5
1.5 Contributions – Interaction . . . 6

2 Prior Art in 3D Interfaces 7

2.1 3D Interaction Techniques for 2D Devices . . . 8
2.1.1 3D Positioning Techniques . . . 8
2.1.2 Spatial Techniques . . . 8
2.1.3 Perspective Techniques . . . 9
2.1.4 3D Orientation for 2D Devices . . . 11
2.1.5 Summary of 2D Techniques . . . 12
2.2 Interaction Techniques for 3D Devices . . . 12
2.3 Immersive 3D Design . . . 13
2.3.1 Clark's System . . . 13
2.3.2 3DM . . . 13
2.4 Non-Immersive Design . . . 14
2.4.1 Exact Match Systems . . . 14
2.4.2 Relative Locator Systems . . . 16
2.4.3 Two-Handed Input . . . 16

3 Prior Art in Free-Form Surface Design 18

3.1 Parametric Design by Control Point Manipulation . . . 18
3.2 Space Deformation . . . 19
3.2.1 Free-Form Deformation . . . 20
3.2.2 Extended Free-Form Deformation . . . 21

3.2.3 Direct Free-Form Deformation . . . 21
3.2.4 Space Deformation . . . 22
3.2.5 SCODEF . . . 23
3.3 Physics-Based Models . . . 23

4 Two-Handed Refining Editor **26**
4.1 Modeling Interface . . . 26
4.2 The Bat . . . 27
4.3 Model Structure . . . 28
4.3.1 Surface Edits . . . 30
4.3.2 Hierarchical Edits . . . 31
4.4 Surface Display . . . 33
4.4.1 Ruler Texture . . . 34
4.5 Right Hand Operations . . . 34
4.5.1 What Control Points Can Be Chosen? . . . 36
4.5.2 Geometry of Control Point Selection . . . 37
4.5.3 Probe Orientation . . . 40
4.6 Left Hand Operations . . . 41
4.6.1 Constraint Mode . . . 42
4.6.2 Selecting Refinement Level . . . 43
4.6.3 Selecting Reshape Operators: The Ring Menu . . . 43
4.6.4 The Sundial Menu . . . 46
4.7 Device Ergonomics . . . 51
4.7.1 Device Fusion . . . 52
4.8 Sample Surface . . . 52

5 Experiment **54**
5.1 Experimental Approach . . . 54
5.2 Competitive Interfaces . . . 55
5.2.1 Mouse-Based Interface . . . 55
5.2.2 Alias Clone Details . . . 56
5.2.3 One-Handed Interface . . . 60
5.3 Tasks . . . 61
5.3.1 Task 1 . . . 61
5.3.2 Task 2 . . . 63
5.4 Experimental Structure . . . 65
5.5 Subjects . . . 66
5.6 Materials and Equipment . . . 67
5.7 Experimental Steps . . . 67
5.7.1 Survey Part 1 . . . 67
5.7.2 Demonstration of Interface A . . . 69

5.7.3 Practice Task 1 with Interface A . . . 70
5.7.4 Perform Task 1 with Interface A . . . 70
5.7.5 Task 1 Using Interface B . . . 71
5.7.6 Survey Pages 5 Through 8 . . . 71
5.7.7 Task 2 Using Interface B . . . 73
5.7.8 Task 2 Using Interface A . . . 74
5.7.9 Survey Pages 9 Through 12 . . . 74
5.8 Summary . . . 75

6 Experimental Results – Demographics and Training **77**
6.1 Demographics . . . 77
6.1.1 Experience Questions . . . 78
6.2 Times . . . 79
6.2.1 First Session Training Times . . . 79
6.2.2 First Session Practice Times . . . 81
6.2.3 Second Session Training Times . . . 81
6.2.4 Second Session Practice Times . . . 83
6.2.5 Third Session Training Times . . . 84
6.2.6 Third Session Practice Times . . . 85
6.2.7 Fourth Session Training Times . . . 86
6.2.8 Discussion of Training and Practice . . . 87

7 Experimental Results – Pain and Fatigue **90**
7.1 Head and Torso Fatigue . . . 91
7.2 Head and Torso Pain . . . 93
7.3 Left Arm Fatigue . . . 94
7.4 Left Arm Pain . . . 97
7.5 Right Arm Fatigue . . . 97
7.6 Right Arm Pain . . . 98
7.7 Pain and Fatigue Discussion . . . 99

8 Experimental Results – Preferences **101**
8.1 Familiarity . . . 101
8.2 Familiarity Discussion . . . 103
8.3 General Preferences . . . 104
8.4 General Comparative Preferences . . . 105
8.4.1 AutoCAD Comparative Preferences . . . 106
8.4.2 Experimental Interface Comparative Preferences . . . 107
8.5 General Comparison Discussion . . . 108
8.6 Detailed Comparative Preferences . . . 110
8.6.1 Vertex Selection Preference . . . 110
8.6.2 Moving With Constraints Preference . . . 110

8.6.3 Activating Rulers Preference . . . 112
8.6.4 View Control Preference . . . 112
8.6.5 Examining Surface Preference . . . 113
8.6.6 Examining Rulers Preference . . . 114
8.6.7 Menu Appearance Preference . . . 115
8.6.8 Menu Selection Preference . . . 116
8.7 Detailed Comparison Discussion . . . 116
8.7.1 Discussion of Mouse Preferences . . . 117
8.7.2 Discussion of Two-Handed Preferences . . . 118

9 Experimental Results – Task Times **119**
9.1 Trial Results for 3x3 Grid . . . 120
9.1.1 First Block Task 1 Performance . . . 124
9.1.2 MouseOne and MouseTwo Task 1 Performance . . . 124
9.1.3 OneTwo Condition Task 1 Performance . . . 126
9.2 Trial Results for Folding a Box . . . 127
9.3 Summary . . . 131

10 Summary and Conclusions **132**
10.1 Interface Evaluation . . . 132
10.1.1 Times . . . 133
10.1.2 Learning . . . 133
10.1.3 Pain and Fatigue . . . 134
10.1.4 Preferences . . . 135
10.2 Interaction Style . . . 135
10.3 The Thesis . . . 136
10.4 Future Work . . . 138
10.5 Conclusions . . . 139

Bibliography **141**

A Pain and Fatigue Tables **148**
A.1 Head and Torso Fatigue . . . 148
A.2 Head and Torso Pain . . . 151
A.3 Left Arm Fatigue . . . 154
A.4 Left Arm Pain . . . 158
A.5 Right Arm Fatigue . . . 160
A.6 Right Arm Pain . . . 163

B Tutorial For Alias and Two-Handed Users **167**
B.1 Preface . . . 167
B.2 Introduction . . . 167

B.3 Geometry 168
B.3.1 Surface Display 168
B.3.2 Selection 169
B.3.3 Reshaping 169
B.3.4 Reshaping with Orientation 170
B.3.5 Constrained Reshaping 170
B.4 Mouse-Based Interface 170
B.5 Two-Handed 3D Tracker Editor 172
B.5.1 The Right Bat 173
B.5.2 The Left Bat 174
B.5.3 Viewing The Surface 175
B.5.4 The Sundial Menu 175
B.5.5 Menu Items 175
B.5.6 Level Window 176
B.5.7 Hints 177

C Survey Sheets **178**
C.1 Survey Sheets 178

List of Tables

6.1 List of handedness decile ranks and their populations. 77
6.2 Demographic breakdown of subjects by condition. 78
6.3 Summary statistics outlining the subjects' experience. 78
6.4 Stage-by-stage breakdown of experimental time spent. 79
6.5 Mean times spent in experiment. 80
6.6 First session training times reported by interface. 80
6.7 First session practice times reported by interface. 81
6.8 Interface B training times reported by interface accounting for interface presentation order. 81
6.9 Second session training times grouped into familiar and unfamiliar classes. 82
6.10 Second session training times for familiar interface B, broken down by interface type. 82
6.11 Second session practice times reported by interface accounting for presentation order. 83
6.12 Second session practice times – familiar vs. unfamiliar. 83
6.13 Third session training times reported by interface. 85
6.14 Training session 2 compared to session 3. 85
6.15 Third session practice times reported by interface. 85
6.16 Practice session 2 compared to session 3. 86
6.17 Fourth session training times reported by interface. 86
6.18 Training session 3 compared to session 4. 87
6.19 A summary of training times separated into appropriate classes for each session. . . 88
6.20 An estimate of component times that contribute to the total training time. 88
6.21 A summary of practice times separated into appropriate classes for each session. . . 89
6.22 An estimate of component times that contribute to the total practice time. 89

7.1 All-subject fatigue statistics for head and torso. 91
7.2 Midway fatigue comparisons between conditions for head and torso. 92
7.3 Midway fatigue comparisons between conditions for head and torso. 93
7.4 All-subject pain statistics for head and torso. 94
7.5 All-subject fatigue statistics for the left arm. 94
7.6 Null hypothesis tests for all-subject left arm fatigue scores. 94
7.7 Fatigue statistics for the left arm for the MouseOne condition. 95
7.8 Testing null hypothesis for left arm fatigue in the MouseOne condition. 95

7.9 Fatigue statistics for the left arm for the MouseTwo condition. 95
7.10 Testing null hypothesis for left arm fatigue in the MouseTwo condition. 96
7.11 Fatigue statistics for the left arm for the OneTwo condition. 96
7.12 Testing null hypothesis for left arm fatigue in the OneTwo condition. 96
7.13 Midway fatigue comparisons between conditions for the left arm. 97
7.14 End fatigue comparisons between conditions for the left arm. 97
7.15 All-subject pain statistics for the left arm. 97
7.16 All-subject fatigue statistics for the right arm. 98
7.17 All-subject pain statistics for the right arm. 98
7.18 Midway pain comparisons between conditions for the right arm. 98
7.19 End pain comparisons between conditions for the right arm. 99

8.1 All-subject familiarity ratings. 102
8.2 All-subject midway familiarity ratings displayed by condition. 102
8.3 All-subject analysis of variance by condition for midway familiarity. 103
8.4 Pairwise comparisons of midway interface familiarity ratings. 103
8.5 All-subject End familiarity ratings displayed by condition. 103
8.6 All-subject analysis of variance by condition for End familiarity. 104
8.7 Pairwise comparisons of End interface familiarity ratings. 104
8.8 All-subject single-ended preference ratings. 104
8.9 Preference ratings for One and Two-Handed in OneTwo condition. 105
8.10 All-subject single-ended preference ratings grouped from all conditions. 105
8.11 All-subject double-ended comparisons. 106
8.12 All-subject aggregated AutoCAD comparisons. 107
8.13 Aggregate rankings of interfaces versus AutoCAD. 107
8.14 General double-ended comparisons with null hypothesis tests. 108
8.15 Aggregated comparisons between Interface and Not-Interface. 108
8.16 Interface vs NotInterface with null hypothesis tests. 109
8.17 All-subject double-ended comparisons of vertex selection. 110
8.18 Null hypothesis tests for vertex selection preference. 111
8.19 All-subject double-ended comparisons of moving with constraints. 111
8.20 Null hypothesis tests for moving with constraints preference. 111
8.21 All-subject double-ended comparisons of activating rulers. 112
8.22 Null hypothesis T-Tests for activating rulers preference. 112
8.23 All-subject double-ended comparisons of view control. 113
8.24 Null hypothesis tests for view control preference. 113
8.25 All-subject double-ended comparisons of examining surface. 113
8.26 Null hypothesis T-Tests for examining surface preference. 114
8.27 All-subject double-ended comparisons of examining rulers. 114
8.28 Null hypothesis T-Tests for examining rulers preference. 115
8.29 All-subject double-ended comparisons of menu appearance. 115

8.30 Null hypothesis T-Tests for menu appearance preference. 115
8.31 All-subject double-ended comparisons of menu selection. 116
8.32 Null hypothesis T-Tests for menu selection preference. 116
8.33 Summary of preferences for the various interface components. 117

9.1 Trial times in seconds for 3x3 grid – Mouse Interface. 120
9.2 Trial times in seconds for 3x3 grid – One-Handed Interface. 121
9.3 Trial times in seconds for 3x3 grid – Two-Handed Interface. 122
9.4 Statistical tests, block 1 of grid trials. 124
9.5 Statistical tests, block 2 of grid trials excluding the OneTwo case. 125
9.6 Mean difference between minimum trial 1-5 time and trial 6 time. 125
9.7 Minima for both task 1 blocks without second OneTwo block. 126
9.8 Statistical tests, block 2 of grid trials in the OneTwo case. 126
9.9 Minima for second OneTwo block compared to all others. 127
9.10 Trial times in seconds for Box folding – Mouse Interface. 128
9.11 Trial times in seconds for Box folding – One-Handed Interface. 129
9.12 Trial times in seconds for Box folding – Two-Handed Interface. 129
9.13 Statistical tests, all box trials. 130
9.14 Minima for both task 2 blocks. 131

A.1 Null hypothesis tests for all-subject head and torso fatigue scores. 148
A.2 Fatigue statistics for head and torso for the MouseOne condition. 148
A.3 Null hypothesis tests for head and torso fatigue in the MouseOne condition. 149
A.4 Fatigue statistics for head and torso for the MouseTwo condition. 149
A.5 Null hypothesis tests for head and torso fatigue in the MouseTwo condition. 149
A.6 Fatigue statistics for head and torso for the OneTwo condition. 150
A.7 Null hypothesis tests for head and torso fatigue in the OneTwo condition. 150
A.8 Midway fatigue comparisons between conditions for head and torso. 150
A.9 Midway fatigue comparisons between conditions for head and torso. 151
A.10 All-subject pain statistics for head and torso. 152
A.11 Null hypothesis tests for all-subject head and torso pain scores. 152
A.12 Pain statistics for head and torso for the MouseOne condition. 152
A.13 Testing null hypothesis for head and torso pain in the MouseOne condition. 152
A.14 Pain statistics for head and torso for the MouseTwo condition. Caption 153
A.15 Testing null hypothesis for head and torso pain in the MouseTwo condition. 153
A.16 Pain statistics for head and torso for the OneTwo condition. 153
A.17 Testing null hypothesis for head and torso pain in the OneTwo condition. 154
A.18 Midway pain comparisons between conditions for head and torso. 154
A.19 End pain comparisons between conditions for head and torso. 154
A.20 All-subject fatigue statistics for the left arm. 155
A.21 Null hypothesis tests for all-subject left arm fatigue scores. 155

A.22 Fatigue statistics for the left arm for the MouseOne condition. 155
A.23 Testing null hypothesis for left arm fatigue in the MouseOne condition. 156
A.24 Fatigue statistics for the left arm for the MouseTwo condition. 156
A.25 Testing null hypothesis for left arm fatigue in the MouseTwo condition. 156
A.26 Fatigue statistics for the left arm for the OneTwo condition. 156
A.27 Testing null hypothesis for left arm fatigue in the OneTwo condition. 156
A.28 Midway fatigue comparisons between conditions for the left arm. 157
A.29 End fatigue comparisons between conditions for the left arm. 157
A.30 All-subject pain statistics for the left arm. 158
A.31 Null hypothesis tests for all-subject left arm pain scores. 158
A.32 Pain statistics for the left arm for the MouseOne condition. 158
A.33 Testing null hypothesis for left arm pain in the MouseOne condition. 159
A.34 Pain statistics for the left arm for the MouseTwo condition. 159
A.35 Testing null hypothesis for left arm pain in the MouseTwo condition. 159
A.36 Pain statistics for the left arm for the OneTwo condition. 159
A.37 Testing null hypothesis for left arm pain in the OneTwo condition. 160
A.38 Midway pain comparisons between conditions for the left arm. 160
A.39 End pain comparisons between conditions for the left arm. 160
A.40 All-subject fatigue statistics for the right arm. 161
A.41 Null hypothesis tests for all-subject right arm fatigue scores. 161
A.42 Fatigue statistics for the right arm for the MouseOne condition. 161
A.43 Testing null hypothesis for right arm fatigue in the MouseOne condition. 161
A.44 Fatigue statistics for the right arm for the MouseTwo condition. 162
A.45 Testing null hypothesis for right arm fatigue in the MouseTwo condition. 162
A.46 Fatigue statistics for the right arm for the OneTwo condition. 162
A.47 Testing null hypothesis for right arm fatigue in the OneTwo condition. 163
A.48 Midway fatigue comparisons between conditions for the right arm. 163
A.49 End fatigue comparisons between conditions for the right arm. 163
A.50 All-subject pain statistics for the right arm. 164
A.51 Null hypothesis tests for all-subject right arm pain scores. 164
A.52 Pain statistics for the right arm for the MouseOne condition. 164
A.53 Testing null hypothesis for right arm pain in the MouseOne condition. 164
A.54 Pain statistics for the right arm for the MouseTwo condition. 164
A.55 Testing null hypothesis for right arm pain in the MouseTwo condition. 164
A.56 Pain statistics for the right arm for the OneTwo condition. 165
A.57 Testing null hypothesis for right arm pain in the OneTwo condition. 165
A.58 Midway pain comparisons between conditions for the right arm. 165
A.59 End pain comparisons between conditions for the right arm. 166

List of Figures

4.1 The THRED User Interface. 27
4.2 The button-enhanced Bat, with four postures for holding it. 28
4.3 A single square and its refinement into four sub-squares. 29
4.4 A single parent quad before and after reshaping. 32
4.5 Ruler texture applied to the edges of a quad. 35
4.6 A condition in which cracks may appear in the surface. 37
4.7 An isodistance plot for Euclidean, Liang, and Probe metrics. 39
4.8 Line and plane constraint screenshots. 42
4.9 The ring menu. 44
4.10 The sundial menu. 47
4.11 A hierarchical sundial menu. 49
4.12 Photograph of the author using THRED. 53

5.1 Task 1 starting configuration and after two X and two Y subdivisions. 62
5.2 Task 1 after moving columns 62
5.3 The completed task 1. 63
5.4 Task 2 after deleting corners and folding one flap 64
5.5 Task 2 after folding the top and side flaps 64
5.6 The completed task 2. 65
5.7 A pair of rating scales for pain and fatigue. 69
5.8 A rating scale for comparison between two interfaces. 72

B.1 A single quad and its refinement into four sub-quads. 168
B.2 A condition in which crackes may appear on the surface. 169
B.3 View control icons in the window title bars. 171
B.4 Line and Plane constraint screenshots 173
B.5 The Sundial menu 176
B.6 The button-enhanced Bat, and four postures for holding it. 177

Chapter 1

Introduction

The thesis of this work is that for 3D computer-aided design of polygon-based free-form surfaces, two hands are better than one. I show this in three steps: I built a prototype two-handed 6 degree-of-freedom (DOF) tracker-based computer-aided design system called THRED. I then built two other interfaces on top of THRED's geometric editing subsystem: a mouse-based interface and a one-handed 6 DOF tracker-based interface. Last, I performed an experiment that asked subjects to perform a simple editing task with two of the three competing interfaces. The experimental results demonstrate that users preferred THRED, experienced low pain and fatigue with THRED, and quickly performed the tasks with THRED. THRED users outperformed users of the other systems on a 2D surface editing task, and performed as well as mouse-based system users on an 3D box-folding task.

The design of free-form surfaces such as terrain and other natural objects is difficult and time-consuming. Part of the reason for this is that the task domain is 3D space, and designing 3D objects is difficult in general. Traditional CAD systems based on 2D I/O devices such as mice, tablets and video displays make the design task still more difficult because these devices each have one degree of freedom less than the design domain. Broadly speaking, the two areas in which traditional CAD systems fall short are in the perception of 3D shape (output), and in the direct manipulation of 3D data (input).

On the output side, the advent of graphics computers capable of rendering more than 500,000 texture-mapped triangles per second allows the possibility of real-time animation of complex 3D objects. Texture mapping and animation give vital information about the 3D shape of an object, and their intelligent deployment can help the user to better understand the 3D shape of the objects being edited.

On the input side, 3D objects can be more easily manipulated by 6 DOF tracking devices than with a mouse [75], because the user does not have to mentally break down the 6 DOF task into a sequence of 2D operations. Using a 6 DOF device allows the user to directly manipulate the objects of interest without intermediate steps. A 6 degrees-of-freedom device reports position (3 degrees of freedom), and orientation (3 degrees of freedom).

Another reason that free-form surfaces are hard to design is that they are complex. A polygonal

mesh of n vertices contains nominally $3 \times n$ degrees of positional freedom, which must somehow be entered and adjusted by a designer. However, not every vertex needs to be exactly placed from scratch: the designer may be able to broadly state the gross shape, and then refine the areas of the surface that have more detailed features. A hierarchical approach therefore seems in order: The designer first quickly sketches the broad features, and then adds details at finer levels of scale.

1.1 The New Interface

This document presents the design and implementation of a two-handed 3D design system called THRED (Two Handed Refining EDitor) that allows for rapid sketching and refinement of polygonal free-form surfaces. In this modeler, the main input devices are a pair of small 6 DOF trackers with buttons, one manipulated by each hand. The user sits in front of the screen of a high-performance graphics workstation and interacts with the model on the screen. This setting allows the user to access the traditional mouse and keyboard input devices as events warrant. The type of surface being created is a hierarchical quadrilateral mesh, in which each quadrilateral (quad) can be refined into four sub-quads. Each subquad vertex can be moved about and further refined to achieve the desired shape.

Because both hands hold a 6 DOF tracker while using THRED, the traditional mouse-based interaction techniques are not readily available without the right hand releasing its grasp on its tracker. A collection of interaction techniques were developed that allow the 6 DOF trackers to pick vertices on the surface, to move vertices, to select commands, and to control the viewing and rendering of the surface.

The theory guiding this work is that people work best when they are able to use both hands in complementary roles. Guiard [28] gives psychophysical evidence for the idea that the left and right hands quite often act as elements in a kinematic chain. For right-handed people, the left hand acts as the base link of the chain. The right hand's motions are based on this link, and the right hand finds its spatial references in the results of motion of the left hand. Also, the right and left hands are involved in asymmetric temporal-spatial scales of motion (right hand for high frequency, left hand for low frequency). THRED uses this natural division of manual labour by assigning the (low-frequency) setting of spatial context to the left hand, and the (high-frequency) selection and picking operations to the right.

1.2 Evaluation of the Interface

THRED has three major parts:

- The interface.
- The surface data structure.
- The surface operations.

Each of these parts can be changed without significantly altering the structure or the interface to the other parts, which means that different interfaces to the same data structure and geometric

operations can be validly compared. With THRED, one can compare interface styles without confounding factors, because they are based on the same underlying operations, and are performing the same fundamental task.

Comparisons of user interface styles and techniques often lack a stable basis of comparison because the interfaces being compared are interfaces to two different programs, with different functionality. Because of this difficulty, interface studies often concentrate on small, easily-described tasks, such as picking, menu selection, and so on. The drawback with this concentration on minutia is that one may miss the big picture. That is, each of the little interface parts may be well characterized, but the application that integrates them may not work well at all. Conversely, techniques that work only moderately well in isolation may fit together more effectively because the drawbacks of each of the techniques are hidden by the actions of the other techniques.

With this study, three interfaces to THRED's underlying database and operations were built. The first interface is of course the two-handed interface, with a 6 DOF tracker for each hand. The second interface uses only one 6 DOF tracker, which leaves the other hand free to enter commands at the keyboard or to use the mouse. This one-handed system is similar to Liang's JDCAD [41, 42]. The third interface uses the traditional mouse and keyboard, in which the mouse does geometric manipulations and menu selection. The mouse-based system has a look and feel similar to the Alias modeler [1]. With these three interfaces, the comparison has a stable basis, because the operations and the data structures are the same for each.

A user test of the three interfaces was conducted to evaluate their effectiveness and ease of use. A total of 18 students from a first course in AutoCAD learned two of the three editing styles in practice sessions, and performed a total of 20 trials – 10 on each interface.

In the first task, they were instructed to create a simple regular 3x3 grid from a single square. The simplicity of the object means that the subjects had no difficulty in understanding exactly what to create. The second task required them to fold up a box using the 3x3 grid.

In addition, subjects were asked to fill in a questionnaire outlining their experience, their preferences, and their level of pain and fatigue.

1.3 The Thesis

The thesis of this work is that for 3D computer-aided design of polygon-based free-form surfaces, two hands are better than one. The research plan is as follows. I built a prototype two-handed 6 DOF tracker-based computer-aided design system called THRED. I then conducted an experiment that compared subject's performance on THRED, a clone of a commercial CAD package, and a one-handed version of THRED.

This document will show that the THRED is as easy to use as a well-designed mouse-based CAD system. It will also show that despite the experience advantage that the average user has in mouse-based systems, and with only a small amount of training, users can become as proficient in the use of THRED as with mouse-based systems. It will demonstrate that THRED can outperform a mouse-based system on a simple 2D task, and that THRED's impact on user pain and fatigue is negligible.

This document will also show that compared to a one-handed 6 DOF tracker based system, THRED is faster, much more strongly preferred, and less fatiguing to use. The central claim of this work is that THRED is as effective as mouse-based systems of similar complexity for 3D tasks, and is more effective than mouse-based systems for 2D tasks. This claim is supported by the experimental results.

For two-handed interfaces, the experimental results support the idea that the hands should be given complementary roles. Although no other two-handed style was compared, the equal or superior performance of THRED at least indicates that this organization schema is not a poor one.

Chapters 2 and 3 outline the prior art in 3D user interfaces and free-form surface manipulation, respectively. Chapter 4 reports the structure and content of THRED. Chapter 5 explains the experiment, including details of how the Mouse-based and One-Handed editors work. Chapter 6 reports the demographic data of the subjects, and analyzes the training and practice times. Chapter 7 reports and analyzes the pain and fatigue survey results. Chapter 8 reports and analyzes the preference and familiarity survey results. Chapter 9 reports and analyzes the timing results for the trials.

Chapter 10 will discuss the generalizability of the thesis statement, and contains conclusions and recommendations for future work.

1.4 Contributions – User Study

This document presents a contribution to the state of knowledge about 6 DOF tracker-based interfaces. This knowledge arises from a user study that trained the experimental subject in two of three competing user interface styles – THRED, a mouse-based system with a look and feel similar to the Alias modeler [1], and a one-handed system similar to Liang's JDCAD [42, 41]. In this experiment, a survey of the subject's prior computer experience was conducted, and the subject was asked to perform repeated trials of two simple tasks. Subjects also reported their subjective preference and familiarity with the competing interfaces, and rated their levels of pain and fatigue throughout the experiment.

This is an unbiased framework for the evaluation of user interfaces, because the underlying database, graphics and editing operations are the same for all three editors. Subjects perform the same task in each interface, issuing substantially the same commands in the same order. The result is that the difference in performance arises purely from the differences in the user interface.

Subjects are randomly assigned to perform the tasks on two of the editors, the presentation order of the editors is balanced to avoid practice effects, and the survey questions are balanced in presentation order as appropriate.

Taken together, this experimental schema represents a new direction in user testing. The integration of trial times, preference ratings, and pain and fatigue ratings is entirely new. Additional information collected during the trial also allows an estimate of user training and practice times. This testing schema succeeds at characterizing the user interface in the large. The experimental results are as follows.

1.4.1 Learning

Subjects had at least 3 months of formal training in AutoCAD, an average of 1.9 years mouse experience, and no 6 DOF tracker experience. The Mouse-Based subjects therefore have an advantage in terms of familiarity and expertise.

As a result training and practice times on the bat-based systems took 16 minutes longer for only the first task. Immediately after this first training and practice session, One- and Two- handed users were able to outperform the Mouse-Based users on average. This is an important result, because it indicates that only a small amount of extra training on THRED is sufficient to redress the balance in familiarity between THRED and mouse-based interfaces. This extra time is spent on introducing the new interaction techniques and having the user get used to manipulating the trackers.

1.4.2 Times

The subjects' trial times indicate that on a strictly 2D task, THRED users can outperform users of the Mouse-Based and One-Handed systems. Considering the much higher level of familiarity that subjects had with mouse-based systems, this is a strong result. That is, despite the slowness of the 6 DOF trackers, users are able to outperform a mature well-established, well-understood technology on its home turf – 2D editing. This is a new finding, since other user studies of 6 DOF trackers exclusively concentrate on 3D tasks.

For the 3D task, the Mouse-Based system and the Two-Handed system performed about equally well. It is clear that the Mouse-Based system performance was possible because the task devolved to a series of rote operations in each of the 3 orthographic views.

1.4.3 Pain and Fatigue

This is the first survey of pain or fatigue effects related to the use of 6 DOF tracking systems outside the study of simulation sickness. The results of this survey indicate that THRED presented minimal difficulties in the realm of user pain and fatigue, while the One-Handed interface induced statistically significant fatigue.

This is important because a common assumption about 3D user interfaces is that they are ergonomically poor. However, the pain and fatigue ratings show that THRED caused as little increase in pain and and fatigue as the Mouse-Based system.

1.4.4 Preferences

Finally, in the survey of overall interface preference, subjects preferred THRED the most, and the One-Handed interface the least. Nowhere in the general or the detailed questions of the preference survey is there evidence to suggest that subjects preferred the One-Handed interface over THRED. The Mouse-Based system was rated between THRED and the One-Handed interface in the general ratings, but some detailed rankings yielded higher Mouse-Based preferences.

The results of the preference survey yield new knowledge about the advantages and disadvantages of the mouse-based systems over THRED. In particular, although THRED's new sundial menu is an improvement over existing 6 DOF tracker-based techniques, it has some ways to go

before it is competitive with mouse-based menus. The trial times also indicate that the 3-view drawing technique used by most CAD systems offers some capabilities that are worth investigating in THRED.

1.5 Contributions - Interaction

THRED is the first example of a 2-handed free-form surface editor. It is the only system that treats the left hand as more than just a holder for the coordinate frame. It is only the third example of a 2-handed system that uses any 6 DOF tracking technology [53, 30]. The two-handed version of Poston and Serra's Virtual Workbench [58] appeared after THRED was first published [60].

Based on a buttonless commercial 6 DOF tracking system, I introduce a compact 3-button enhancement that is light and easily manipulated by the user's fingertips.

THRED is the testbed for new interaction and visualization techniques that can be applied to other 3D interfaces that use one or more free-space 6 DOF trackers like the Polhemus Isotrak.

The new visualization techniques are:

- Combined rendering of random and contour textures on a surface to provide the user with improved depth perception of the scene.

- A repeating ruler texture along polygon edges that allows users to visually take measurements of edge length by visual inspection. If the user desires, the entire surface can be textured in this way, allowing hundreds of measurements to be available simultaneously.

The new interaction techniques are:

- The sundial menu is an orientation-based circular menu technique. The menu allows users to select objects and text items, and is hierarchical, potentially allowing thousands of menu choices.

- The probe selection technique allows the user to pick both surface features and objects that are small and far away. It is an improvement on Liang's spotlight [42, 41] in that the ambiguities of the spotlight are eliminated, and there is a user customization that makes the probe easier to use.

- Two-Handed constrained manipulation uses the left hand to constrain the motion of the right hand's manipulation of the surface. This allows the user to state a parameter with each hand. The left hand chooses the axis or plane that the right hand will move along, and the right hand selects the distance to move. The technique is natural for inexperienced users to pick up, and in user tests, some subjects were able to overlap left and right hand motion. This technique allows for the statement of 9 degrees of freedom in one operation.

Chapter 2

Prior Art in 3D Interfaces

The current state of the art in commercial CAD systems uses a graphics workstation, with a high-resolution display for output and a keyboard, a mouse (or digitizing tablet) and perhaps a dial box for input. Such systems have the advantage that the hardware is inexpensive and easy to obtain. Software developers can therefore rely on the presence of this hardware, and can concentrate their efforts on improving the efficacy of their design software.

The fundamental operations that these input devices must perform are *positioning*, *selecting*, *entering text* and *entering numeric quantities* [22, 23, 24]. Clearly, the keyboard is the device of choice for entering text. For entering numeric values, either the keyboard or any number of traditional interaction techniques (sliders, dials) may be used.

A mouse can be used quite effectively for 2D positioning and 2D selecting, but mice lack 1 degree of freedom (DOF) to specify 3D positions. Moreover, mice can specify neither 2D nor 3D orientations directly, and therefore require some sort of interaction technique to translate 2 DOF input into 3D orientation.

A single 2D pick operation can be used to select a 3D point by projecting the point onto the projection plane. If N selectable 3D points project to the same location on the 2D screen, then the 2D pick operation must be followed by some other operations that disambiguate between these N points to give the final selection of the 3D point. 3D picking is therefore a simpler task than 3D positioning, because the picking algorithm need only select from a list of candidate points.

The upshot is that for 3D input, a collection of interaction techniques are required to translate the 2D locator's data into 3D position and/or orientation. Another alternative is to use a 6 DOF locator, which specifies the 3D position and 3D orientation of the *sensor* part of the locator device. Some 3D trackers report just 3D position, and so have only 3 DOF. In the following sections, a 6 DOF tracker corresponds to a device that reports 3D position and orientation, while a 3 DOF tracker reports just position.

The location of the input device is represented by on the display a cursor that moves in tandem with the device. 2D devices like the mouse or tablet have a 2D cursor which always appears on top of all the other items in the display. A 3 DOF tracker is represented by a 3D cursor that is a three-dimensional rigid object in the 3D scene. The 3D cursor can be hidden by scene objects. A 3D cursor representing a 3 DOF position tracker changes its 3D position in tandem with the

tracker, but maintains a static orientation throughout. A 3D cursor representing a 6 DOF position and orientation tracker changes both its 3D position and orientation in tandem with the tracker.

2.1 3D Interaction Techniques for 2D Devices

This section reviews the interaction techniques that have been developed to allow 2D devices such as the mouse of the tablet to be used to state 3D position and 3D orientation.

2.1.1 3D Positioning Techniques

For 3D positioning of an object, only the position is being specified. 3D orientation remains fixed. The basic idea underlying all 2D interaction techniques for 3D positioning is to fix one or two DOFs while the remaining DOFs are manipulated by the mouse. Usually, the X and Y axes of the mouse correspond to two of the three canonical axes of the design arena, or they correspond to two of the major axes of an object being designed.

There are two broad classes of techniques. One class subdivides the 2D screen space and assigns a particular axis or axis pair to each spatial region. The region that the cursor is in when a mouse button is pressed determines what axis values are to change. In its orthographic views, Alias modeler uses the left mouse button to change bothe the vertical and horizontal degrees of freedom, the middle mouse button to change just the vertical, and the right button to change the horizontal.

The other class assigns an axis or axis pair to a mouse button or a combination of buttons. For example, button 1 is pressed for movement in the XY plane, button 2 for the YZ plane, and button 3 for XZ. Another common example, used in the perspective view of the Alias modeler, is to use button 1 for X axis motion, button 2 for Y and button 3 for Z. Another variation is to allow two buttons to be pressed (a *chord*) to indicate the two axes to change. Positioning is therefore done by creating a point at some sensible default location, and then pressing the appropriate mouse buttons to drag it to the desired location.

Each class offers complementary advantages and disadvantages. The spatial techniques require that the user think more carefully about where the mouse cursor is placed, while freeing mouse buttons for other tasks. Conversely, the button-assignment techniques use more buttons, but allow more freedom in mouse placement. This may enlarge effective target size in some applications, resulting in faster picking speeds [21]. The rest of this section discusses the various techniques of spatial subdivision for 3D positioning.

2.1.2 Spatial Techniques

The most common spatial technique is taken from drafting, in which the three orthographic views of top, side, and front are presented. Some systems also present a perspective view in the remaining quadrant of the screen [1]. Each orthographic view rectangle defines a region in which the mouse's XY plane maps directly to the view's orthographic axes. The technique is conceptually easy to understand and quite easy to implement, and is familiar to anyone who has studied drafting. Another advantage of the technique is that the orthographic drawings are unambiguously metric, so the user can measure precisely where the cursor is in the design space by examining where the

mouse cursor is on the screen. Grids are often drawn to help this, and the coordinates of the mouse in the current orthographic drawing are often printed in a status bar.

There are three main disadvantages to the three-view technique. First, 3D positioning requires that the mouse be moved between two widely separated points on the display. This lack of compactness tends to result in long interaction times, because each time a third dimension is needed, the mouse must be moved more than half the screen height into another drawing. Following this large mouse movement, a small target must be picked, which according to Fitts' Law [21] results in long picking times. However, for picking already-established points, the user can usually get away with picking in only one drawing if there is only one 3D point at the 2D location on the screen.

Second, orthographic drawings make it difficult to learn the 3D structure of the object being designed. Some systems allow each drawing to be panned and zoomed separately, which makes 3D perception more difficult because the user is not even looking at the same objects at the same scale. Perspective projection is a good depth cue for scenes with parallel lines and/or right angles [38], and the 3-view technique restricts a perspective view to only 1/4 of the screen. Some systems allow picking in the perspective view, but disambiguation must occur in the orthographic views, and the positioning of new points also requires added orthographic input.

The main problem with this technique is that so much of the screen is wasted on views that are hard to visually understand, making the 3D perception of the model difficult.

Sittas [64] used three orthogonal projection planes behind the object of interest to facilitate 3D positioning. A 3D point can be defined by selecting two points on two different projection planes. Also, a finite size construction plane can be used to construct curves and other shapes on a plane not parallel to any axis planes.

Ken Herndon and others at Brown University [29] have a similar technique called *interactive shadows* in which the three planes behind the object each show an orthographic projection (shadow) of the object. The user can select the object in the perspective view or on one of the shadow objects. Motion in the shadow views is constrained to the projection plane. These two techniques make use of the unused background space behind the perspective view of the objects of interest, but like the three-view technique, the specification of a 3D point requires that the user move the mouse to two widely separated areas of the screen.

2.1.3 Perspective Techniques

To eliminate wasted orthographic views, one single view must be used, in which the scene is typically rotated by the user to allow different views of the objects. This view is often drawn in perspective. The cost of the single-view strategy is that a 3D point must be stated in a series of steps, and a number of interaction techniques have been developed which specify a mouse syntax for these steps.

Nielson and Olson developed a technique called the *triad mouse* in which mouse movement is used to determine the the nearest canonical axis [46]. The axis whose projection is closest to the velocity vector of the mouse is used as the axis to operate on. Since the movement is always along one of the axes, simultaneous translation in all three directions is impossible.

The triad mouse separates mouse motion into six imaginary pie-shaped regions emanating from the starting point of mouse movement. The pie-shaped regions are small if two axes project very close together, so the user must carefully select the direction of mouse motion to coincide with the desired axis.

Maarten van Emmerik [72] developed a similar technique for 6 DOF manipulation in which a 3D axis is presented which has seven control points (handles) for user manipulation. The handles are at the positive and negative axis endpoints, plus the origin. If the user drags the origin handle, the action is interpreted as translation along the axis whose projected direction is closest to the direction of the dragging, just as with the triad mouse. If the user drags one of the axis end handles along its corresponding axis, the action is interpreted as a scaling along that axis. Otherwise, it is interpreted as a rotation around one of the other axes whose projected direction is closest to that of the dragging. Each of the three basic transformations (translation, rotation, and scaling) can be performed as a direct manipulation. However, all the transformations are still limited to along/around the three major axes.

The van Emmerik technique allocates spatial subregions using a *widget*, or auxiliary object, which has *hot spots*, *pick regions*, or *affordances* that have special meaning when picked. Eric Bier [5] developed a pair of auxiliary objects called the *skitter* (a 3D cursor) and the *jack* (a 3D coordinate system). The 3D position of the skitter is determined by the intersection of the nearest surface in the scene with a selection ray specified by the 2D cursor. The skitter's coordinate system (orientation) conforms to the coordinate system of this surface. A jack can be placed at the current skitter position and can be used as an anchor for geometric operations. The motion of a 3D point can also be constrained, either to a line or to a plane in 3D. The specification of relative position and orientation between a pair of objects can be specified using two jacks, each associated with one of the objects. In this way, it is very easy to put two mating faces against each other. This technique relies on existing objects in the scene. Some other techniques must be used to create 3D objects from scratch and to place objects in arbitrary 3D positions and orientations.

Another auxiliary-object technique called the *rack* was developed at Brown University [65]. The rack is a kinematic chain of thin bars with spherical pick regions at the endpoints. A rack is attached to an object, and each bar has special semantics associated with it, such as *stretch*, *twist* or *bend*. Unlike the auxiliary objects mentioned earlier, the rack is not rigid, and higher-level modeling semantics are attached to each of the pick regions.

Most of these techniques are useful, but they have two common features. First, simultaneous translation along more than two degrees of freedom is impossible because only two degrees of freedom are available from the mouse. Most of the auxiliary-object techniques (except Beir's skitters and jacks) further restrict motion to one degree of freedom at a time. The upshot is that 3D positioning is time-consuming, as the user must decompose 3D point entry into a set of 1 DOF actions. When fine adjustments are required, constrained manipulation of the object helps the user to focus on the dimension being changed. However, if quick sketches are desired, breaking the 3D point entry task into 1 DOF or 2 DOF sub-tasks (such as creating a rectangle, and then add the third dimension to make a 3D box), forces the designer to think in terms of subspaces, rather than focus on the 3D

nature of the object [42].

Second, specification of 3D orientation is not well-supported. All of the techniques listed here require that orientation be stated as a sequence of 1 DOF rotations about some axis. Most techniques restrict the choice of axes to the canonical axes of the design space. If an auxiliary object can be attached to a design object, then the object's axes can be used as rotation axes, or the surface of an object can be used as a rotation plane. The breakup of 3D orientation into a sequence of 1 DOF axis rotations makes the specification of 3D orientation especially difficult because single-axis rotations are not commutative and are somewhat convolved with each other. In most situations, this means that one cannot progressively fix each single degree of freedom, because subsequent rotations about any axis disturbs the orientation about all the axes.

Another problem with the auxiliary object techniques is that each auxiliary object takes up a nontrivial amount of screen space, and if auxiliary objects can be added to the scene (as in Bier's and van Emmerik's systems), they could add a to screen clutter. One way to deal with this is to hide them when they are not needed, which leads to the problem of appropriate hiding policy, and requires that the user remember hidden system state. The reduced visual clutter can therefore come at the expense of added cognitive load.

2.1.4 3D Orientation for 2D Devices

Another way of dealing with 3D orientation is to apply a mapping of mouse movement that closely resembles real-world concepts. In a relative motion technique developed by Evans, Tanner and Wein [19], three successive mouse positions are compared to determine whether mouse motion is linear or rotary. Linear horizontal or vertical motion controls rotations about the vertical or horizontal axes, linear diagonal movement rotates about both the vertical and horizontal axes, and rotary movements control rotation about the axis pointing out of the screen.

In the virtual sphere method developed by Chen [14], the user manipulates a superimposed 3D trackball in an absolute fashion as though it were real. Chen [14] performed an experiment to compare these two approaches, which resulted in a user preference for the virtual sphere, but no performance difference.

Shoemake [62] designed a similar technique called the Arcball, which is based on the use of quaternions in specifying a 3D rotation. The user is presented with an imaginary sphere displayed as a circular disc. The user first picks a point on the surface of this imaginary sphere and a great arc is drawn from the initial point and the current point. The sphere pivots at its center around the axis of the great arc and the resulting rotation (with its magnitude doubled) is taken as the output. The starting point and end point determine the rotation angle as well as the rotation axis, which is not restricted to the three major axes X, Y, and Z. The manipulation is also "memoryless" in that the result is fully determined by the starting point and end point, and ignores intermediate points of the arc.

The advantage that the Arcball has over the virtual sphere is that there is a fairly intuitive mapping between mouse movement and resultant orientation, because of Arcball's lack of memory. This means that slightly different mouse strokes do not result in somewhat different orientation

changes, as in the virtual sphere. Also, an entire revolution can be stated in a single arcball mouse stroke, while the virtual sphere can only perform half a revolution about the horizontal and vertical axes.

These virtual ball techniques can only be used for 3D orientation. Position must be entered using another technique, which means that the user must break down the task into positioning and orienting subtasks.

2.1.3 Summary of 2D Techniques

The basic problem with using 2D devices for 3D CAD is that there are not enough degrees of freedom available to state 3D position and orientation. A constraint maintenance system that uses information about the objects being edited can powerfully supplement 1 DOF and 2 DOF operations because features of the object in the scene can be used to state the line or plane in which motion is to occur. Bier's *snap-dragging* system is an example [5].

In the absence of constraint maintenance, each task must be broken into a series of two or more 2 DOF or 1 DOF positioning or orienting subtasks. Sometimes this is acceptable if the user is interested in adjusting a single degree of freedom of an object, such as the X position or the Y rotation. In the early stages of design, however, gross placement and orientation ought to take place quickly and naturally, but the breakup of 3D placement into many steps slows down quick placement. More insidiously, breaking 3D placement into a series of 1 DOF tasks tempts the user into making inappropriately fine adjustments in one dimension before the object has been grossly placed in the remaining two dimensions. The user can invest a lot of effort in refining elements of a design that may be ill-conceived and may have to be discarded.

Because of a natural human tendency to preserve one's time investment, this may result in the user committing to a bad design. In the 2D Graphical User Interface (GUI) world, anecdotal evidence shows that sketches are much more useful in design reviews than a more finished looking design. Wong [76] found that rough sketches of 2D GUIs kept her team from talking about unimportant low-level details, while "finished" looking interfaces caused them to talk more about the "look" rather than interaction issues. The belief that colleagues give more useful feedback when evaluating interfaces with a sketchy look is commonly held in the design community [52]. In the field of graphical design, Black's user study [6] found that "the finished appearance of screen-produced drafts shifts attention from fundamental structural issues". The implication of these studies of the design process is that presenting too much polished detail at the beginning of the design cycle will seduce the designer into thinking that the functionality is good, and will tend to result in poor design functionality.

2.2 Interaction Techniques for 3D Devices

To get around the problem of 2D devices forcing the user to break up 3D manipulation tasks, some CAD systems that use non-traditional I/O devices such as 6 DOF trackers and Head-Mounted Displays (HMDs) have been built. The previous non-traditional CAD systems that use 6 DOF input devices can be subdivided into two broad categories - immersive and non-immersive.

2.3 Immersive 3D Design

The immersive style uses a HMD to display the scene to the user. Each eye in an HMD views a graphics display through a lens system that is worn on the head. There is a display for each eye, mounted binocularly in such a way that the whole display system can be directly and easily moved by the user. The user's view direction is continuously updated by a position and orientation tracking system connected to the head-mounted display.

The advantages to the HMD are:

1. Stereopsis is delivered directly by the display hardware.
2. The display can freely move about the room, so the user can look around objects and explore the geometry of the virtual world.
3. Users can turn their heads to view more of the scene, allowing as much of 90% of the sphere surrounding the user to be visible.

To manipulate objects in an immersive system, the user handles a 6 DOF position and orientation tracker or a DataGlove [79]. Each 6 DOF tracker controls the position and orientation of a 3D cursor which can select objects, move them around, and otherwise edit and examine the objects of interest. The following sections review CAD systems that use the immersive style.

2.3.1 Clark's System

In 1976, James Clark [15] built a system which used a HMD and a single 3 DOF "wand" with some buttons to design free-form bicubic patches. The user edited the surface by moving the control points of the patch. The system used the new control point locations to update the shape of the mesh in real time. To select a control point, the user moved the wand within the selection radius of the desired point, pressed the wand button, and dragged the control point to the desired position. There was no menu system nor any other function selection mechanism for the wand, so the user was limited to one primitive operation. A see-through HMD was used, allowing the possibility of commands to be entered at the keyboard.

2.3.2 3DM

More recent work at the University of North Carolina and Chapel Hill resulted in 3DM [10], an immersive system in which the user manipulates a 6 DOF Polhemus tracker with two buttons mounted inside a billiard ball.

The system was loosely based on MacDraw, where the user picks operations and geometric primitives from a rectangular menu in 3-space containing an array of menu options. Menu picking was done by moving the 3D cursor within the 3D bounding box of the menu item of interest and pressing a button. The menu panel also had a series of pull-down menus containing other operations not directly available on the main panel.

To create triangles, the user clicks a button for each of the vertices. Boxes, cylinders and the like by can be created by dragging out the bounding box. A simple constraint maintenance system provided feature-based snapping for triangle vertices.

The user selected existing objects by placing the 3D cursor within the object of interest and pressing a button. The user could then select an operation from the menu to scale, rotate, translate or otherwise modify the object. Object color could be changed by picking a new color from a planar rectangular color menu in 3-space.

The various modes in the system are signified by a unique cursor shape. For example, when "fly" mode is selected, the cursor changes to an airplane and the user can move around in the virtual world.

In 3DM, positioning and orienting of objects can be done directly by grabbing the desired object with the 6 DOF tracker and moving it directly to the desired position and orientation. Indeed, this is the most common interaction technique for 6 DOF trackers – drag the object and simultaneously change its position and orientation. An alternative to flying through the world provided by 3DM is the option of grabbing the entire world and moving it as if it were one single object.

2.4 Non-Immersive Design

Instead of using a HMD, the *non-immersive* style displays its information on a display mounted in a fixed location. 3D perception may optionally be aided by stereo glasses of some sort.

2.4.1 Exact Match Systems

Exact match systems are built around the idea that user performance is maximized by constructing the user's workspace and the visual output so that the user's spatial inputs correspond exactly to the visual space presented on the display. The display is viewed as a visual-only extension of the real space that the user is working in, or in some cases, the user works directly within the displayed visual volume.

Christopher Schmandt [55] built a system called the *interactive stereoscopic computer graphic workspace* which used PLZT shutter glasses to provide a binocular stereoscopic display to the user. A half-silvered mirror was used to overlay the graphics scene on top of a small real-world work area. The user manipulated a Polhemus-tracked wand in this workspace, and could trace out curves and polygonal paths. Because the workspace was behind the mirror and due to bad ergonomics, users had to stretch out their arms, and they reported that their arms got tired after a short period of use.

Poston and Serra [49] at ISS in Singapore have recently built a similar system, using Crystal Eyes shutter glasses and a full mirror. This system therefore does not suffer Schmandt's problem of a visual mismatch between real and virtual objects. The input device is a 6 DOF joystick called the Immersion Probe which the user manipulates like a pen. A Polhemus 6 DOF tracker is also used. Poston and Serra have been careful with the ergonomics of the workspace, allowing users to rest their elbows on the desk of the workspace. Similar to 3DM, a cursor-in-volume test is used for all of the main interaction techniques [58]. General-purpose techniques such as menus and sliders are provided. The main menu is located the bottom of the workspace. Its interaction is simplified by using only the horizontal degree of freedom of the input device once the cursor is within the total menu volume. There are also application-specific techniques such as a stereotactic head frame for

neurosurgery.

Iwata [32] placed a hand similarly behind a mirror to share space with a scene, but the emphasis there was on force feedback to the fingers, rather than to manipulate a tool. The hand manipulates the scene by grabbing a force-feedback hand master grip, constraining the hand to a very small work volume.

In these three systems, the mirror makes the virtual and real workspaces coincide, maximizing the kinesthetic correspondence between hand input and visual scene output. This correspondence occurs in translation, orientation, and scale, thereby enhancing dexterity. Assuming that the user is working behind the plane of projection, the mirror is a necessary component of an exact match system. A mirror is not sufficient, however. If the control-to-display (C/D) ratio is not unity, or if there is a translation between the sensor position and the cursor position, there is not an exact match.

Deering [18] built a system with similar dexterity goals in which the computer generated scene in the display was carefully adjusted to match the real scene. Sources of distortion such as CRT curvature were accounted for in the 3D geometry processing, and a head-mounted tracker was used to determine the user's view position and orientation. Deering's system did not use a mirror, so the user interacted with the display *in front of* the plane of projection. Deering demonstrated a virtual lathe in which a Polhemus-tracked knife carved away material to form objects of revolution. There was also a circular menu system called the "fade-up" menu. The items of a fade-up menu are laid out in concentric circles, with more frequently used items on the inner rings. Selection is accomplished by moving the cursor within the volume of the desired menu item. Deering's system has the problem that the user's hands can obscure part of the scene on the display, unlike the mirror-based systems.

The mirror solves arm fatigue by visually moving the workspace to the top of the desk. The mirror also bends the optical path so that the hands are out of the line of sight. Two issues with the mirror are that it requires special furniture, and tasks like text editing or programming require text to be painted backwards on the display.

Maintaining an exact spatial match means that the input devices can only be used as absolute locators, because otherwise at least a translation is required from the cursor position to the tracker position. This implies that the workspace available to the user is the minimum of the display's visual volume and the tracker's volume. If a subset of the input volume is difficult to reach, it means that the corresponding subset of the screen is difficult to reach. One benefit of an absolute locator is that certain regions of the real work volume can be dedicated to system control, much like fixed menus on 2D tablets.

The basic premise of these "exact match" systems is that users will perform better the more exact the match between the real and the virtual spaces. They seek to eliminate the sensory conflict between proprioception of the workspace and vision of the workspace. More research is needed to determine how much conflict between these two sensory systems will cause a breakdown of user performance.

2.4.2 Relative Locator Systems

In contrast to the exact match idea, systems that use relative location techniques for spatial input allow for the idea that the input space and the visual output space may differ in some easily-understood way. One typical mapping is to allow the user's input to be affected by a translation and a scaling before being drawn on the screen. Most systems provide a re-origining command to move the cursor to a new location, allowing the user to move the input device to a more comfortable position and still not lose access to the entire visual space. This is similar to the act of picking up the mouse to move it to a new location on the mouse pad.

Jiandong Liang at University of Alberta built a system called JDCAD [41, 42] which uses a single Polhemus sensor to create 3D mechanical parts. The user creates items by selecting a primitive from a Polhemus-based ring-shaped menu and dragging out the bounding box. The user can then select objects, reshape them either freely or along a constrained axis, and group them together. JDCAD provides a number of editing and alignment operators such as center-to-center alignment, center-to-edge, and so on. JDCAD also has CSG capability, allowing the user to create complex objects by performing union, intersection or subtraction between two objects.

Object selection can be controlled by a *spotlight*, in which the 6 DOF cursor is changed into a little flashlight from which a translucent cone projects, representing the cone of light. A spotlight that has identical visual geometry to the translucent cone is attached to the cursor. As the user sweeps the flashlight about the scene, the objects that fall within the cone intersect visually with the cone wall, and are highlighted by the spotlight. The nearest object to the center of the beam is highlighted as the candidate item. This technique is similar in spirit to a laser beam selector, except small or far away objects can be easily picked.

Emanuel Sachs [53] and his colleagues at MIT built a system called 3-Draw, which was a two-handed system for sketching 3D drawings. The left hand held a lightweight clipboard with a Polhemus tracker attached, and the right hand held a Polhemus 1-button stylus. The left hand held the base coordinate frame of the model, so as the the left hand moved, the model moved in synchrony on the screen. The right hand directly selected the data items to be manipulated, and selected the operations to be performed using a screen-aligned 2D menu controlled by the 6 DOF tracker. The fundamental data items were polylines and points, which were created by sweeping out the curve to be drawn with the right hand. The stylus was also used to pick and directly manipulate points and polylines. Point-to-point distance tests were used for interaction with the objects, and the 2D menu was controlled by projecting the 3D cursor to the projection plane and performing the interaction in 2D. The simultaneous use of two sensors takes advantage of people's innate proprioceptive knowledge of where the two hands are in space.

2.4.3 Two-Handed Input

Buxton et al have been doing work on two-handed input for a number of years. For selecting a text object in a document, Buxton and Myers [11] compared a one-handed mouse-based technique using a standard scrollbar to a two-handed technique that used the left hand to scroll. The left hand manipulated a touch pad. The two-handed system was found to be significantly faster.

More recently, Buxton has been involved with a two-handed technique called Toolglass, in which the left hand controls a semitransparent menu that is clicked through by the right hand. Bier's paper [4] lists numerous example Toolglass applications. None of these examples deals with stating constrained motion of the mouse cursor in any way.

Finally, Kabbash, Buxton et al. have performed experiments examining the performance of either hand on some standard 2D devices [37], and comparing Toolglass to tear-off menus [36].

The only work that Buxton has done with 3D input devices per se is with Zhai in his investigation of the "Silk Cursor" [77].

Hinckley et al. [30] built a two-handed 3D neurosurgical visualization system. Like 3-Draw, the left hand holds the brain model represented by a ball, and the right hand manipulates a cutting plane represented by a flat plate connected to a second Polhemus tracker. The user moves the cutting plane to interactively generate cutaway views of the brain. New users immediately understand how the manipulation works, and need essentially no training to start doing useful work.

Chapter 3

Prior Art in Free-Form Surface Design

Free-form surfaces are used in many important areas of design such as in the shipbuilding, automotive and aircraft industries, and in entertainment (special effects and cartoons). A significant amount of research effort has been expended on this topic, yet it is still a challenge [45] [33].

Two main approaches have been taken to representing surfaces in 3-space [20]: The parametric representation maps the 2D domain (u, v) onto 3-space using three sets of equations: $(x(u, v), y(u, v), z(u, v))$. The parametric approach is usually associated with spline functions, and has been quite successful in the commercial realm [1].

The implicit representation defines a surface as a set of points

$$(x, y, z) \text{ such that } F(x, y, z) = 0$$

Solid modeling is philosophically close to this approach, but there has been no real success to date in finding techniques for designing free-form surfaces [45].

3.1 Parametric Design by Control Point Manipulation

Since a parametric surface patch is defined by a set of basis functions and a set of control points, an obvious way of interactively changing its shape is to manipulate the control points. Usually, the surface being designed is displayed, along with its associated control mesh. For some functions, the control points and the surface do not coincide. This lack of coincidence makes it difficult for the user to understand how adjusting a control point will affect the shape of the surface. Much time can be wasted experimenting with the wrong control point or collection of control points. Also, because there are typically a large number of control points associated with the surface, the scene can become quite cluttered.

To deal with these problems, Forsey and Bartels [25] developed a direct manipulation method for editing B-spline surfaces. The user is presented with a collection of pseudo control points (edit points) on the surface that can be picked and manipulated. When the user moves an edit point, the surface is reshaped to maintain its contact with the point. An edit point is a point on the surface that is maximally influenced by a control point, so for each control point, the corresponding edit

point can be found and highlighted on the surface. By recursively subdividing a surface patch, one can modify the shape of the surface at various levels of detail. The advantages with this system are that the radius of influence of a edit point is easy to see, and the user gets the impression that the surface is being directly manipulated.

The drawbacks of Forsey and Bartels' system are that it is limited to tensor product splines, and cannot accommodate surfaces of arbitrary topologies. Also, their system allows only one control point to be moved at a time, which makes the editing process rather tedious. Arbitrary points on the surface cannot be edited, although they do state that some sort of root-finding process could be used to find the control points that maximally influence an arbitrary surface point.

Following on this idea, Fowler [26] developed a tensor product spline surface editor that allowed the user to edit any point of interest on the surface. The system would find the control points that had maximal influence on this point, and the user could change the position, tangent plane and direction and twist vectors of that surface point. The desired surface changes were linearly mapped to changes in the four nearest control points. Because of this representation, only local changes can result. Fowler noted that his system did not allow for non-local changes due to the maximal-influence mapping.

3.2 Space Deformation

Alan Barr [2] originated the idea of using a deformation of 3-space to obtain curved objects. The deformations he describe correspond to the metalworking operations of bending, tapering, and twisting about an axis. These operations can be combined to form complex shapes derived from quite simple building blocks such as boxes and cylinders.

Transformation matrices with entries containing functions of the deformation parameter are used to describe these operations. For example, bending about a line parallel to the X axis and along the Y axis is accomplished by varying a bend angle parameter θ along Y. Outside the bend region, θ is constant at either its minimum or maximum value. The center of the bend is along the line $[s, y_0, 1/k]$, where s is the parameter of the bend axis. The center of the bend is at $y = y_0$, and the radius of curvature of the bend is $1/k$ measured in radians per unit length.

Within the bend region, $\theta = k(y - y_0)$, and the X, Y and Z values have the form:

$$X = x$$

$$Y = -(z - \frac{1}{k})sin\theta + y_0$$

$$Z = -(z - \frac{1}{k})cos\theta + \frac{1}{k}$$

The Jacobian is used to determine the transformation of the surface normal and the surface tangents at each point. The Jacobian is also a function of the deformation parameter, and is therefore different for each vertex.

The advantage of this idea is that it fits well with standard geometric operations on 3D points and is reasonably intuitive. These operations can be composed, since applying one deformation yields a solid that can be deformed by another operator. Snibbe et al. [65] built a 3D deformation

widget called the *rack* which could be attached to an object in order to deform it. The rack's handles are manipulated in order to specify the deformation of the object.

3.2.1 Free-Form Deformation

Following on this work, Sederberg and Parry [56] developed a method of defining a Free-Form Deformation (FFD) of solids. Instead of using a vocabulary of elementary deformation operators, the FFD approach defines a volume of space that is to be deformed and a function that moves a vertex in the starting space to a point in the destination space. In this case, the deformation vocabulary is the tensor product trivariate Bernstein polynomials. The starting space is a 3D parallelepiped (box) that is the control lattice of the Bernstein polynomial, and the destination space is the convex hull of the the control lattice after the control vertices have been moved.

To generate the deformation, each point x on the object to be deformed that lies within the starting box is transformed into the parametric coordinates (s, t, u) of the box:

$$x = x_{origin} + s\vec{S} + t\vec{T} + u\vec{U}$$

where $\vec{S}$, $\vec{T}$, and $\vec{U}$ are the parametric basis vectors of the box. The deformed points are then generated by evaluating the vector valued trivariate Bernstein polynomial:

$$x_{ffd} = \sum_{i=0}^{l} \binom{l}{i} (1-s)^{l-i} s^i \left[\sum_{j=0}^{m} \binom{m}{j} (1-t)^{m-j} t^j \left[\sum_{k=0}^{n} \binom{n}{k} (1-u)^{n-k} u^k \; P_{ijk} \right] \right]$$

The P_{ijk}'s are the control points of the control box, which are initially defined as

$$P^0_{ijk} = x_{origin} + \frac{i}{l}\vec{S} + \frac{j}{m}\vec{T} + \frac{k}{n}\vec{U}$$

These trivariate Bernstein volumes generate a mapping from $R^3 \Rightarrow R^3$ which preserves derivative continuity within the control box. Sederberg and Parry point out that each face of the control box maps into tensor product Bezier surface patches, and each edge of the control box forms a Bezier curve. The FFD volume is therefore a Bezier volume, and the usual facts about Bezier continuity apply. For example, the boundary of the control box will yield a C_0 discontinuity if the boundary vertices are moved, so one can add an extra boundary control box for each level of continuity required. This is a straightforward extension of the 1D and 2D Bezier curve and surface properties.

The same approach, using B-spline volumes, was implemented by Griessmair and Purgathofer [27]. The advantages of these techniques are that the deformation mathematics are well-characterized, and these deformations can be applied to any surface representation method. Also, a given deformation lattice can be applied to any object or collection of objects as if it were a shaping tool. This reusability is attractive because a deformation can be designed and tested on simple objects, then applied to complex objects.

Both systems require that the user design a set of control points which define the deformation, and one problem with this is that the deformation control points are not related to the objects being deformed. Like designing parametric surfaces by manipulating control points, the FFD volume and its embedded object is edited by manipulating control points that lie outside the object of interest.

Another problem with this technique is that the control lattice has the topology of a cube, so in order to create a deformation of arbitrary shape, a surrounding box of FFD control points must be created that matches the desired topology. For example, radially symmetric deformations are difficult to create with this technique.

3.2.2 Extended Free-Form Deformation

Further work by Coquillart [16] extended the FFD idea by allowing control meshes of arbitrary shape. Three modifications are made to the FFD idea:

First, a composite EFFD lattice is composed of a piecewise collection of degree 3 Bezier *chunks*, or $4 \times 4 \times 4$ boxes of control points. The EFFD lattice is created by face-to-face *welding* or mating one chunk to the next. Face-to-face matching control points are moved and edited together in tandem. For example, to mate the chunks A and B along an XY face, vertices P_{ij3} of chunk A and Q_{ij0} of chunk B always occupy the same points for $0 \leq i, j \leq 3$. Tangent continuity is maintained by maintaining collinearity of the face neighbor control points (P_{ij2} and Q_{ij1} for example) across a weld. Because of the piecewise point and tangent welding, only P_{333} of each chunk can be edited, but both control point position and tangent can be edited by the user.

To create a non-box-shaped EFFD volume, points along a chunk face can be welded to each other, creating a wedge shape. The other free faces can be welded to other chunks to create various prisms. Another elementary chunk in Coquillart's system is a cylinder, because a circular chunk can only be exactly represented by rational splines.

Unlike FFD, EFFD is conceived as a surface editing tool, so a composite EFFD lattice is first placed about the surface to deform. Each point on the surface must go through two steps to determine its predeformed location. The first step finds which chunk the surface point is in, and the second step evaluates its location in the chunk's parametric space.

The composite lattice can thus approximate the shape of the desired deformation without the need for extra vertices in order to satisfy the requirements of the tensor product representation. Coquillart's Extended FFDs are therefore able to more easily create effects such as star-shaped reliefs and fist-shaped dents in objects.

Joy [34] has a similar approach to trivariate modeling called the parametric hyperpatch. The parametric hyperpatch has some higher-level editing operations.

These methods can express a wide variety of deformations, but they all have the drawback of being an indirect manipulation technique, because the deformation acts as a tool on the surface of the objects being designed. The user must first design the tool, then use it to design the object. Moreover, the tool may take some effort to design if a large number of control points are needed, and there is always some danger that the deformation will not act as expected.

3.2.3 Direct Free-Form Deformation

In reaction to this, a direct manipulation approach called DFFD was described by Hsu, Hughes and Kaufman [31], in which the user manipulates the points on the object instead of the points on the trivariate deforming lattice. The matrix of the lattice control points P_{ijk} is displaced by the

equation

$$\Delta Q = B\Delta P$$

where ΔQ is the displacement of n selected points on the object being deformed, B is a matrix composed of B-spline polynomials, and ΔP is the matrix of the control point displacements. The B-spline basis is used because of its local control properties, and its efficient computation with large lattices of control points. To update the deformation, ΔP must be computed, using B^+, the pseudo-inverse of B:

$$\Delta P = B^+\Delta Q$$

Because the pseudo-inverse gives a least-squares solution, the change in control point positions is minimized. For situations where only a few constraint points are moved, the system is underdetermined, and the result is reasonable. If there are too many constraint points, the problem becomes overdetermined, and the pseudo-inverse produces results with large errors. Large errors are a clue that the user should generate a finer mesh of control points, but Hsu et al don't say how one should do this. Another problem with this scheme is that the deformations are B-spline shaped, and the envelope of the deformation cannot be directly controlled. Like FFD, the deformation envelope is a parallelepiped.

DFFD offers the advantages of direct manipulation, and a reduction of the amount of extraneous information presented on the screen during editing.

3.2.4 Space Deformation

Another formulation of the direct manipulation approach has been taken by Borrel and Bechmann [7, 3], in which the user specifies a deformation by selecting a point on the surface and then giving a point in space to which it will be constrained. Their editor DOGME (Deformation Of Geometric Models Editor) deforms space in a 3-step process.

The deformation is performed by a function $d : R^n \rightarrow R^n$. Given a set of object points U, the deformation function $d(U) = \Delta U$. The deformation function $d(U)$ is expressed as the composition of the *parameterizing* function $g : R^n \rightarrow R^p$ and the *extrusion* function $f : R^p \rightarrow R^m$ and a linear transformation $T : R^m \rightarrow R^n$.

$$d(U) = Tf(g(U))$$

The parameterizing function g transforms the Cartesian coordinates of U into a p-dimensional parameter space. The extrusion function f characterizes the shape of the deformation, so for example, local deformation about the constraint point could be obtained using B-spline polynomials. The shape of the extrusion function f is imprinted on the deformed area.

Once f and g have been chosen, the deformation depends on the matrix T, which is computed so as to satisfy the constraints. Like the DFFD system, the pseudo-inverse of T is calculated to solve the system such that

$$T = f(g(U))^+ d(U)$$

Borrel and Bechmann's formulation is a generalization of the process that occurs with EFFD and DFFD. Similar to EFFD's chunk-finding step, the parameterizing function g defines the volume that

is to be deformed and parameterizes the points inside the volume. Like DFFD, a set of constraints are satisfied by transforming the parameter set into a higher-dimensional space and using the pseudo-inverse to change the control parameters. The linear transformation T can be viewed as a projection from $R^m \rightarrow R^n$.

If m is too small for the number of constraints, the pseudo-inverse provides a projection that gives a least squares fit. The advantage of this system is that the point constraints are easy to understand.

Bechmann shows [3] that the DOGME formulation formally contains all of the previous Free-Form Deformation formalisms, since the extrusion function f can represents the shape of the deformation, and g represents shape of the affected space.

3.2.5 SCODEF

To get around the problem of surprising configurations that satisfy the constraints, Borrel and Rappoport [8] introduced a standard deformation called SCODEF (Simple COnstrained DEFormation) which has a desired displacement and a radius of influence. Each constraint point determines a local B-spline basis function centered at the constraint point, falling to zero at points beyond the radius. The displacement of a point is a blend of these basis functions, obtained by a linear combination that ensures that all constraints are satisfied. The advantages with this system are that the deformation is easy to compute and the deformation is easy to understand. A difficulty is that there is no formulation for line or area constraints, so like Forsey's system [25], objects designed by the system look like a collection of bumps.

3.3 Physics-Based Models

Another means of deforming of surface is to simulate some sort of physical process that acts on the surface in a well-understood way. A physics-based model has the advantage that it acts in a realistic way, and the idea is that the designer's experience with real materials can be applied to modeling surfaces.

Demetri Terzopoulos and his colleagues [68, 69, 70, 71] are the main proponents of this approach, in which the surface to be deformed is defined as a rectangular mesh of masses which exert translational and rotational forces on their nearest neighbors and the outside world.

To use this approach, a continuous surface must first be parameterized in 2D, then discretized into a rectangular array of identical masses. The forces that are exerted on these masses are represented by a partial differential equation that accounts for inertia, damping, and the potential energy of the elastic deformation of the body. These forces sum to produce the net externally applied force. External forces such as springs, gravity, viscous flow, friction, or collision forces can be applied to a mass or a collection of masses. This differential equation is also discretized, and then numerically integrated through time for each mass to determine the deformation of the surface.

For perfectly elastic surfaces [69], the elastic component of the differential equation must be determined, and it relies on two material properties. The first property is the material's resistance to stretching, which is zero when a surface is at its rest position, and increases in magnitude as

the distance from the rest position increases. The greater this stretch constant, the greater the resistance to stretching. For each mass, the distance to the neighboring masses minus the rest distance is calculated, and multiplied by the stretch constant.

$$net\ force = \sum_{i \in neighbor\ masses} StretchConstant(Distance_i - RestDistance_i)$$

A large stretch constant could be used to approximate the properties of solid metal, while smaller constants make the surface more like a thin sheet of rubber.

The second elastic property is the material's resistance to bending, which again increases as the surface deviates from rest. In this case, the surface normals at the mass points are compared, and the greater the difference between the neighboring rest orientations, the greater the bending force magnitude. A large bend constant will force an initially flat sheet to stay flat, like paper, while a small bend constant could be used to approximate cloth.

To simulate inelastic materials, the ideal springs are replaced with collections of springs, dampers and slip units. A damper exerts a force proportional to the velocity of motion, and a slip unit stretches or shrinks when the force exceeds the unit's yield force. The addition of a damper in parallel to a spring serves to filter out high-frequency movements, making the motion of the masses more viscous. The slip unit makes the material able to permanently maintain a deformation. For example, metals behave elastically when the applied forces are small, after which they yield plastically, resulting in permanent shape changes. When the yield limit is exceeded, the slip unit changes the effective rest distance for the springs that accompany it.

Given an $M \times N$ array of masses, these force equations result in a $MN \times MN$ matrix called the *stiffness matrix*. Because the surface properties are local, the stiffness matrix is banded and sparse. The inertia and the damping forces are diagonal matrices, so the numerical solution of the system amounts to the repeated solution of a sparse linear algebraic system.

Numerous methods of solving this system can be applied, including Choleski factorization, successive over-relaxation, multigrid, and so on, depending on the size of the model and on how long one is willing to wait. The numerical solution can fail when the springs are too stiff, because the stiffness causes unexpectedly large instantaneous forces to be accidentally applied.

The advantage with this sort of simulation is that the surface deformation looks quite real. If the bending and stretching constants are set appropriately, the force model gives the behavior of a *thin plate surface under tension* [67], which reduces to the traditional *spline under tension* in the case of curves.

The problems with this technique are that it is computationally intensive, it requires that the parameters be tuned appropriately, and there is some danger that the system may not be numerically solvable under the chosen technique. Another problem is that the given formulation seems to require parameterization on a fixed grid, which is not very useful if a hierarchically-refined surface is to be used.

Terzopoulos said in 1988 [68] that a supercomputer might be able to simulate a moderately complex surface (20×20 nodes) in real time. Current hardware is starting to approach these speeds, so real-time editing of a nontrivial dynamic system might be feasible.

For a surface editor using this technique, not all of the realistic material properties are required unless the user has some means of feeling the forces. For example, plasticity can be faked by simply setting the rest distances of the springs to be the current configuration before an editing operation is started. Also, Terzopoulos is interested in realistic animation, which is a stronger condition than using semi-realistic motion to create an object. Perfect rubber sheets may be more than adequate for interactive reshaping of a surface.

Chapter 4

Two-Handed Refining Editor

THRED is based on an SGI Crimson RealityEngine graphics workstation that has a screen, keyboard, mouse, and a pair of Polhemus Isotraks with buttons (Bats) [75], which are the main input devices. Each hand holds one bat, and the user manipulates the graphical scene with them. Each 6 DOF sensor has a distinct role, with the dominant hand being responsible for picking and manipulation, and the less-dominant hand being responsible for context setting of various kinds. For the sake of rhetorical convenience, this document will refer to the dominant hand as the right hand and the less-dominant hand as the left, but the system is ambidextrous, because the Polhemus trackers are symmetric and can be handled with equal ease by both hands.

4.1 Modeling Interface

When THRED starts up, the user is presented with a window containing two subwindows and the top menu bar, as shown in figure 4.1. There is only one pull-down menu in the menu bar, containing traditional file operations such as load, save, and quit.

Below the menu bar is the refinement level window, which displays all of the levels of refinement that have been made to the surface, and shows which levels are currently selectable. The refinement window is only 100 pixels tall, and is used for quick reference and for quick interactions to change the current level of refinement. This window also shows the current reshape operator on the left.

The majority of the screen space is taken up by the main window, which displays the work area and the left and right cursors. Each bat has a corresponding cursor which is continuously updated in the 3D work area, and moves in tandem with the 3D position and orientation of the bat. The two cursors are both transformed from a common room coordinate system [61], so that cursor separation on the screen is the same as the separation of the bats in real space, and the cursors are located in approximately the right place on the screen. The user may request a re-origining operation, which moves both cursors by the same amount so that the midpoint of the line segment between them is at the center of the screen space. The re-origining operation helps the user to find the "sweet spot" where manipulations are the most comfortable.

The right cursor is drawn as a 3D jack, and the left cursor is a jack with a flat plate attached, reminiscent of a clipboard. The cursors change shape depending on the current constrained manipulation mode, as outlined in section 4.6. Once the user understands which cursor belongs to which

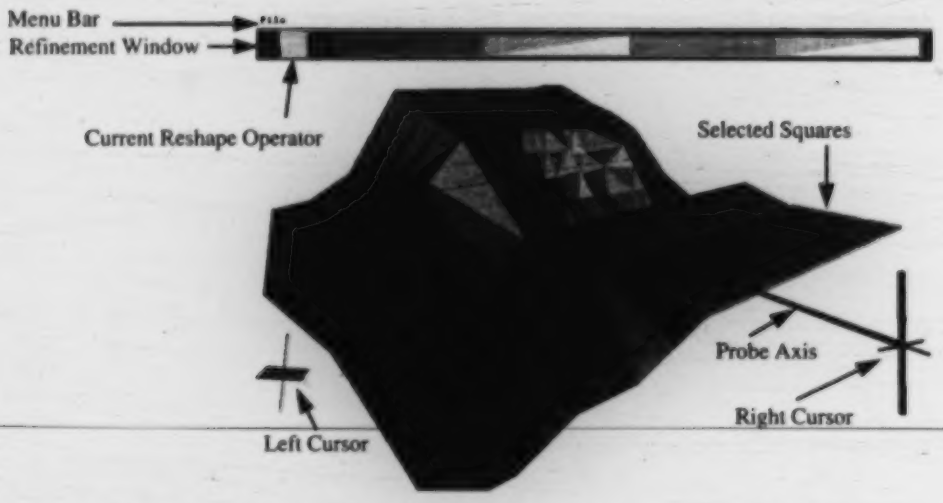


Figure 4.1: The THRED useer interface. The surface is a sketch of the Standish Face of the Sunshine Village Ski resort at Banff, Alberta.

hand, cursor confusion rarely arises because the user relies on proprioception to find his/her hands and therefore their surrogates in the design space.

In figure 4.1, the left cursor is to the bottom left and the right cursor is to the bottom right. THRED is in line constraint mode, and the right cursor is snapped to the vertical axis. Some squares to the right are currently selected, awaiting modification. The level refinement window shows that all levels are selectable, and indicates that THRED is in tandem reshape mode.

4.2 The Bat

The top of figure 4.2 shows one of the bats, which is a standard Polhemus sensor with three Omron JP3101 buttons attached. These buttons are momentary pushbuttons that have an audible and palpable click.

In the usual posture, the thumb can press button 3 (nearest the wire), and button 2 (the middle button) (fig 4.2 bottom far left). Shifting between button 3 and button 2 is a simple matter of rocking the thumb over the pivot of the button casings (fig 4.2 bottom middle left). The index finger presses button 1, which is therefore used for *primary* tasks. The sensor is supported in the fingers by the middle finger and/or the third finger, and the small finger is sometimes used to grip the cable.

Two-button chords using button 1 and either button 2 or button 3 are easy to perform, since the thumb and the index finger each control one button. The middle finger supports the force applied by the forefinger and thumb, and these fingers both move along their prime directions of movement. A chord using buttons 2 and 3 is more difficult because either the thumb must press both buttons, or the index finger must press button 2 using abduction, or scissor-like movement. Three-button chords require a different posture, with the middle finger on button 1 and the third and small finger

Figure 4.2: Top: The button-enhanced Bat. Bottom: Four possible postures for holding the bat. The X axis points up and the Z axis points to the left in the bottom left three pictures.

supporting (fig 4.2 bottom middle right).

This bat is an improvement on commercial products such as the Logitech 2D/6D mouse [43] because of its smaller size and weight. Because the bat is smaller, it can be manipulated by fingertips alone, while the Logitech mouse must be held aloft by the thumb and little finger, with the three middle fingers pressing the buttons. Thus, button presses pivot the mouse so that it is pressing on the palm of the hand, so in effect the entire hand is responsible for positioning and orienting the mouse. As a result, there is no precise coordination between the thumb and the index and middle fingers, which is vital if fine manipulation is to take place [13]. Also, the need to grip the mouse with the whole hand limits the range of orientations to that of the hand, while the fingertip grip allows the user to rotate the bat to a wide range of orientations (fig 4.2 bottom far right). Also, the lighter weight of the bat limits the amount of finger fatigue users experience.

The usual posture for the THRED user (figure 4.12) is to point the X axis of the bat up, as shown on the three bottom left images of figure 4.2. The bat's Z axis points to the left, towards the screen. This pose arises because the user's elbows are resting on the arms of a chair or on the desk, and the weight of the forearm is balanced on the elbow in a minimum-energy posture. Also, the weight of the cable is most easily managed when the cable is hanging straight down. The posture shown in the bottom right image of figure 4.2 is rare, because the weight of the wire makes it more difficult to hold the bat in this position.

4.3 Model Structure

THRED allows the user to create hierarchically-refined polygonal surfaces based on quadrilaterals. The hierarchical refinement allows areas of low surface variation to be represented by a few large rectangles, while areas of fine detail have the appropriate concentration of closely-spaced vertices. Also, broad-scale geometric operations can be performed on a few parent rectangles and then propagated appropriately to the children of these rectangles [25]. These broad-scale edits will leave the fine surface details of the child rectangles mostly intact, except with distortions induced by surface stretch.

At the root level of refinement, the surface is simply a rectangular grid, with each vertex connected to its North, South, East and West neighbors. To polygonize this grid, each rectangle is subdivided into a NorthWest and a SouthEast triangle, allowing a direct edge connection from a vertex to its NorthEast and SouthWest neighbors. Each rectangle can be subdivided into exactly 4 sub-rectangles, resulting in 8 corresponding triangles.

The data structure of THRED is based on the *square* (figure 4.3), which contains one vertex and a set of pointers to all its nearest neighbors in the mesh. The square also contains a list of its four children, surface normals, plane equations, and graphical display list IDs. The two triangles of the square are to the NorthEast of the square's vertex, and the North, East and NorthEast neighbors are used to determine the other 3 vertices of the quadrilateral.

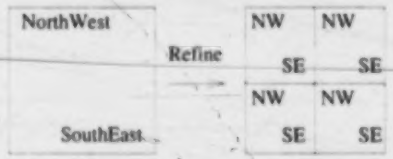


Figure 4.3: A single square and its refinement into four sub-squares.

The data structure is similar to a quadtree [54] with the addition of parent links and links between the nearest neighbors at the same level of the tree. Each of the child squares of a parent square contains one vertex, with the SouthWest child having the same vertex location as the parent.

The entire surface data structure can be thought of as a hierarchy of refined grids. Level 0 (the root level) is the basic rectangular grid with its interconnections and plane equations. Level 1 contains all of the root squares that have been refined at least once. If two level 0 squares are refined and are neighbors, then two of their level 1 children are also neighbors, and will have neighbor connections. For example, if squares **S** and **E** are each refined, and **E** is the eastern neighbor of **S**, then the SouthEast and NorthEast children of square **S** will be neighbors to the SouthWest and NorthWest children of **E**, respectively. This structure continues recursively down to the finest level of refinement.

Aside from bookkeeping modifications, only the following three operations can modify the mesh data. The *refine* operation creates four child squares for a square, the *move* operation changes the vertex data and recalculates other salient geometric information, and the *add row/column* operation adds a row or a column of squares to the periphery of the root-level mesh.

To create a model, the user starts with a single square at the origin and commences adding rows and columns, refining squares, and moving vertices. The user builds the desired surface by iteratively modifying, refining, and examining the surface given the available interface tools. This view of surface modeling is similar to Forsey's Dragon Editor[25], with the obvious exception that the Dragon Editor manipulates bicubic B-Spline surfaces. To get interactive update rates, Forsey approximates the hierarchical surface mesh using a pair of triangles for each square in a manner identical THRED's. Forsey has assured me that any surface generated by THRED that obeys the hierarchical B-Spline subdivision rules would be renderable by the Dragon Editor.

4.3.1 Surface Edits

To reshape a section of the surface, the user selects the vertices to be changed, and then starts a reshaping operation. The two types of reshaping are reshaping with and without orientation. In reshaping with orientation, the selected vertices are moved about in tandem, following the position and orientation of the bat. The selected set of vertices rotates as a rigid body about the centroid of the vertices. In reshaping without orientation, just the bat's position information is used, and the selected vertices are moved according to one of five reshaping operations: *move everything in tandem*, *cone* reshape, *scale* reshape, *scale to median plane*, and *adopt parent*.

The *tandem move* operation is useful for grabbing a block of squares and moving them all at once. This is the translation-only version of reshaping with orientation.

The *cone* reshape operator acts as if a conical tent were being erected centered at the middle vertex. For a vertex in the selected area, this is implemented by multiplying the cursor motion by a linear factor of the distance from the center vertex. The linear factor is determined such that the most distant selected vertex from the center does not change, and such that the center vertex moves in tandem with the right cursor.

The center vertex is determined by taking the middle square within the selected set of vertices, and if there is a tie, taking the North, East, or NorthEast square as appropriate. Taking a vertex as the center of manipulation is necessary because it yields a clear peak when reshape is performed, and because the operation is undoable by moving the center of manipulation back to its original position. Using the arithmetic mean of all points or just the center of the axis-aligned bounding box is not reversible, because the center point changes relative to the vertices.

Movements in the plane have an analogous cone-like effect to movements perpendicular to the plane, and the result is to change the spacing of vertices on the plane.

The *scale* operation scales the distance of the non-center vertices from the center vertex. This operator can be used to exaggerate or minimize the spikiness of the selected features.

A similar operation is *scale to median plane*, in which the median plane of all the selected vertices is calculated, and the reshape operator scales the perpendicular distance of each vertex from this plane. This operator uses 1 DOF to scale the perpendicular distance of each vertex from the median plane. Cursor movement in one direction scales up the perpendicular distance of each vertex, while movement in the opposite direction scales down the perpendicular distance. When the scaling factor reaches 0, all vertices lie in the median plane, This operator is used to scale in a space that represents a major feature of the data.

The *adopt parent* operation causes the selected squares to adopt the geometry of their parent squares. The selected squares realign into groups of four on their parent squares as if the parents had just been refined. The purpose of this operator is to reduce the amount of high-frequency surface detail, and is used as a "start over" operator. This is not under continuous control, and takes place as soon as the user presses the appropriate Bat button.

4.3.2 Hierarchical Edits

Because of the hierarchical refinement structure, the vertex of a square and the vertex of its SouthWest child are required to be in the same location. This property is maintained throughout the data structure by a bottom-up process. If an edited square is the SouthWest child of its parent, the parent is moved to the same location as its SouthWest child. Each SouthWest child propagates its new vertex location to its parent. The three other child squares (NorthWest, NorthEast, SouthEast) do not propagate their new vertex locations to their parent. This process is recursive, percolating SouthWest square changes up from leaf to root.

Any edit to a square causes a recalculation of its plane equations, using its neighbor pointers to get the appropriate data. These plane equations are used to support the hierarchical editing feature. Usually, the leaf squares are selected for edits, but if the user wishes to change coarse features but maintain fine detail, the user may select non-leaf vertices to edit.

When non-leaf vertices are selected and moved, their descendant vertices ride on the surface of the parent quadrilaterals. The first step of this "riding" operation is to store the current value of all vertices that will be affected by the editing operation. Next, the location of the child vertices on the parent quads is determined. There are four cases:

Child	Operation
SouthWest	Equals parent.
SouthEast	Child is located on the parent's SouthEast triangle.
NorthEast	Child is located on the parent's SouthEast triangle.
NorthWest	Child is located on the parent's NorthWest triangle.

Location on the SouthEast triangle is determined by generating a non-orthogonal source coordinate system S with the origin at the parent vertex location. The X axis of S is along the line from the parent's vertex to the parent's East neighbor, with the $(1,0,0)$ position in S equal to the location of the East neighbor. The Y axis of S lies parallel to the line from the East neighbor to the NorthEast neighbor, with the $(1,1,0)$ position in S equal to the location of the NorthEast neighbor. The surface normal of the SouthEast triangle determines the Z axis of S. The non-orthogonality of S arises from the shear in the Y axis with respect to the X axis. In S the XY plane is orthogonal to Z, and the Y shearing occurs strictly in the XY plane.

Figure 4.4 shows the situation in 2D. The source quad has 3 child vertices; SouthEast child A, NorthEast child B, and NorthWest child C. A and B are located on the parent's SouthEast triangle, and C is located on the parent's NorthWest triangle. The Y axis is parallel to the East edge, and the NorthEast neighbor is at $(1,1,0)$. In the Destination coordinate system, the Y axis has been sheared to the left by leftward and upward movement of the NorthEast neighbor vertex, and A and B have moved appropriately. Note that these vertices are still in the same relative positions on the parent's SouthEast triangle after reshaping as before. Similarly, C is in the same relative position on the parent's NorthWest triangle.

When the parent vertex or its neighbors are moved, a destination coordinate system D is determined using the new values of the parent vertices and surface normals. The transformation T from S to D is generated, as if the origins of S and D were the same.

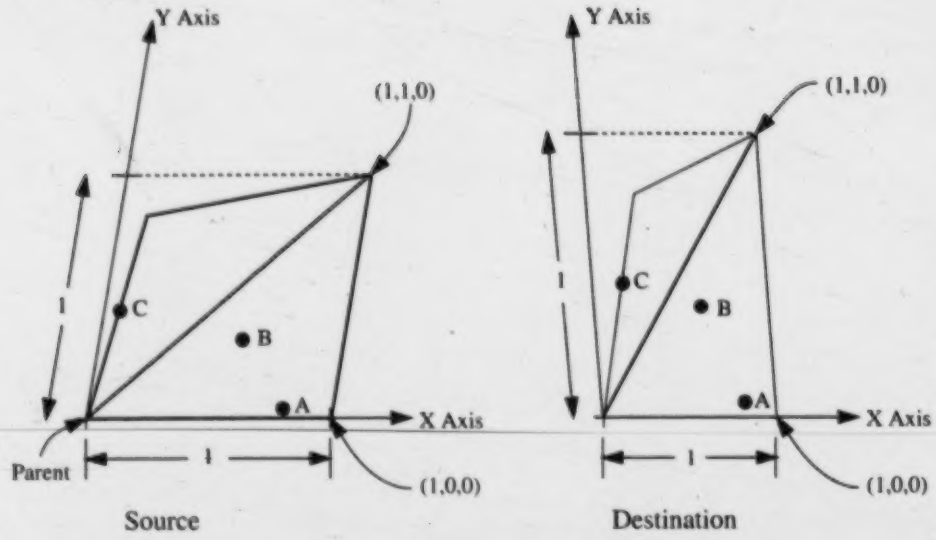


Figure 4.4: A single parent quad before and after reshaping. A, B and C are child vertices that ride on the parent triangle. The child vertices remain in the same relative postions on the parent at the end of reshaping.

$$TS = D$$

$$TSS^{-1} = DS^{-1} = T$$

S will be singular if the three triangle vertices are collinear, in which case T is set to the identity matrix. The origin of D is at the current location of the parent vertex, so to apply the transformation T, the child vertex is first located in the S coordinate system, then T is applied. Next, the child is transformed back into the world coordinates W from the D coordinate system.

$$Child_W - SourceParent_W = Child_S$$

$$T \cdot Child_S = Child_D$$

$$Child_D + DestinationParent_W = Child_W$$

This transformation is applied to the SouthEast and NorthEast children. An analogous operation is performed on the NorthWest child using the parent's North neighbor as the Y axis of the coordinate system, and the line from the parent's North neighbor to the parent's NorthEast neighbor to determine X.

This process is applied to all descendant vertices, using the S and D coordinate systems of the coarse parent triangles.

Briefly, these transformations embed a skewed coordinate system on the triangle in which the legs of the triangle are by definition unit length. The location of the children in this coordinate

system is maintained as the user stretches the surface, and since the triangle's surface normal does not change length, a child's height from the triangle plane is maintained.

4.4 Surface Display

In order to help the user visually locate the control vertices, the NorthWest triangle of a square is drawn a darker shade than the SouthEast triangle. Each level has its own unique hue, so for example the root level is magenta, level 1 is yellow, and so on. For diffuse and specular reflection, the face normal is used, so that the surface looks strongly faceted. This helps the user find each editable vertex at a glance.

By default, only leaf vertices are available for picking. To pick non-leaf vertices for coarse edits, the user must select which level(s) are able to be picked (as explained in section 4.6.2). One of these selectable levels is the finest selectable level F, and the surface display changes to show that vertices at levels greater than F are unavailable. The geometry of the surface remains unchanged, but triangles at level $F + 1$ or more are drawn using the level F color scheme. All descendant triangles that are above the NorthWest triangle are drawn in the dark F color, and all of the SouthEast descendants are drawn in the light F color.

One of the problems in working with free-form surfaces is that it is often difficult to immediately understand their shape. This is a 3D perception problem, and the standard depth cues [9], of hidden surface removal, surface lighting, and perspective projection are of course employed. THRED adds continuous motion parallax and surface texture. Stereopsis is not currently used in THRED [18, 73], although this could be added, and fog (depth cuing) is not used because fog is most useful for distant objects. The continuous motion parallax is controlled by grabbing the model with the left bat and moving it around with the left hand.

Two types of textures are mapped onto the surface of the triangles, a random texture and a contour texture.

The random texture is 64 by 64 pixels, with just an intensity channel. Each pixel in the texture has a random brightness value between 0.8 and 1.0, which is used to modulate the brightness of the underlying polygon color after all the surface lighting and shading has been performed. The texture mapping process maps the repeating random pattern onto each triangle like wallpaper. Rectangular and hexagonal grid patterns were tried for the surface texture but they added visual clutter in the near field of view and created moiré patterns at greater distances.

Mapping the random texture to the surface helps the user to immediately understand the surface shape. In situations where it is ambiguous whether a surface feature is a depression or an outcrop, the surface texture combined with the lighting model usually resolves the problem. This is because the user can use natural perceptual knowledge of texture gradient to understand the surface shape. Random texture also gives the interior area of a triangle a visual feature that can be followed as the surface is being reshaped, so texturing also helps editing.

The contour texture produces a series of equal-height contour lines on the surface, which helps the user gauge the height of features and to get a feel for the relative sizes of features. The texture is a one-dimensional line of pixels with just an intensity channel, and this line of pixels repeats to

form the contour over the entire surface. This texture presents a ruler pattern of meters and tenths of meters on the surface. The tick mark at zero meters uses 4 dark pixels (zero intensity), the 0.5 meter mark uses two dark pixels, and the other 1/10 marks use one dark pixel. The remaining pixels are at full intensity, and the mapping function calculates the distance of the surface vertices from the XY plane modulo the length of the pixel row. The planar interpolation of each polygon handles the rest of the texture lookup procedure. This planar distance function is supplied by SGI's Graphics Library, so only the equations need be specified.

To get both textures on the surface simultaneously, the scene is rendered twice, first using the random texture with the Z-Buffer hidden surface comparison function set to *draw-if-closer*. Then Z-buffer function is changed to *draw-if-equal*, and drawn again using the contour texture and a translucent color.

4.4.1 Ruler Texture

The user may enable selected quads to be drawn with rulers along each triangle edge, as shown in figure 4.5. To draw this, the two base triangles must be subdivided so that there is a unique *ruler quadrilateral* assigned to each edge, upon which is pasted the ruler texture. The SouthWest triangle is replaced with 3 ruler quads and an inscribed triangle, all in the SouthWest triangle's plane. The NorthWest triangle is replaced with a ruler quad on the North and West edges and an inscribed triangle, all in the NorthWest triangle's plane. The diagonal of the NorthWest triangle is already labeled by the SouthEast triangle's diagonal ruler.

All ruler quads are 0.02 meters wide and have a 1 meter long texture with marks from 0 to 10 at 10 centimeter intervals. The 0.5 meter and 1 meter marks are extra heavy to aid visual recognition. The texturing parameters are set up to repeat the ruler texture at the end of 1 meter. As the bottom of figure 4.5 shows, the tick marks are placed on the triangle edge with the numbers directly above, with the exception of 0 and 10 which are to the right and left of their tickmarks respectively. Only one texture is used, so the numbers appear backwards on the North, West and diagonal edges. The zero mark of the South, West, and diagonal rulers are all located at the square's SouthWest vertex. The zero for the North edge ruler is at the quad's NorthWest vertex, and the East zero is at the SouthEast vertex. If an adjacent quad has rulers, then the coincident edges will have a ruler on each side of the edge with the zero mark at the same location.

The inscribed triangle's vertices are calculated by moving each edge towards the center by 0.02 meters. On the SouthEast triangle, each inscribed triangle vertex lies at the intersection point of the two ruler quads incident at that vertex. This intersection is used to draw the inscribed triangle and the ruler quads. Thus, each ruler quad uses 2 outer vertices and the 2 corresponding inscribed vertices. For the NorthWest triangle, the diagonal does not have a ruler, so the North and West rulers are each intersected with the diagonal to find the inscribed vertex locations along this edge.

4.5 Right Hand Operations

The right hand has three major tasks to perform:

- Control point selection and selection adjustment.

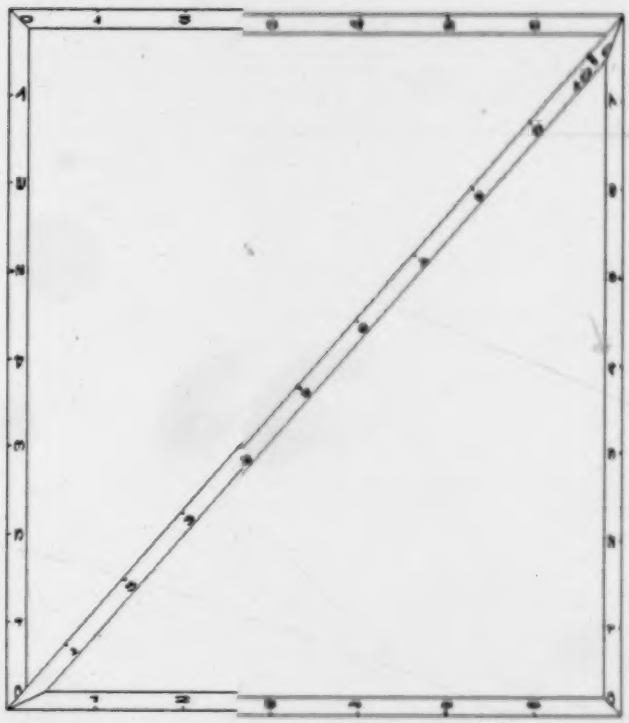


Figure 4.5: Ruler texture applied to the edges of a quad. The lines illustrate how the two base triangles are each replaced with an inscribed triangle and two or three texture-bearing quadrilaterals.

- Reshaping the selected area.
- Reshaping the selected area with orientation.

Button 1 controls control point selection, as outlined in section 4.5.1.

When the user presses right button 2, the squares of the reshapable selected area can be reshaped according to the given reshaping operator. The five reshaping operations are *move everything in tandem*, *cone* reshape, *scale* reshape, *scale to median plane*, and *adopt parent*. Reshaping uses only the right cursor's position data.

When the user presses right button 3, only the tandem reshape mode is active, and the user can directly manipulate both the position and orientation of the selected vertices in tandem. This operation is useful if the user wishes to maintain the shape of a region, but wants to make it steeper or shallower with respect to the rest of the surface.

4.5.1 What Control Points Can Be Chosen?

To reshape the surface, the user must first select which control points or squares are to be changed. This section describes the syntax of control point selection, and the next section describes the geometry of control point and object selection using the right bat.

The user may add or remove vertices from the *Currently Selected Set* S, which is initially empty. To add or remove vertices from S, the user selects a rectangular collection of vertices R, and the new currently selected set S is determined by

$$S \leftarrow S \cup R - S \cap R.$$

To determine R, the user may select one control point, or a rectangular collection of points using the *Press-Drag-Release* syntax. To select one point, right button 1 is pressed and released on the desired control point. To select a rectangle of squares, the user presses and holds button 1 at any control point, drags the right cursor to the opposite corner, and releases button 1. The rectangle may be degenerate: a single row or column of points is valid. When the user presses button 1, R and S are calculated, and every time the user drags the cursor to a new vertex, R and S are recalculated.

The set of squares S is outlined by yellow piping along its periphery. S can be updated and modified by making more rectangular selections.

Selection depends on the current selection context, so for example if the user has restricted selections to level 3 or less, then level 4 and deeper levels will not be selectable. Also, once the user selects a control point of a given level, S cannot include higher or lower level control points. This is accomplished by following the periphery of R as the user is dragging the cursor from the starting point to the current point, and halting progress along the X or Y direction when a different level is encountered.

The selected set S may be refined or reshaped. In cases where the selected area is on the border with a square of shallower refinement, then this edge of the selection area is only refinable, not reshapable. Without this restriction, the user could grab a vertex that lies on the midpoint of

a less-refined edge. When this vertex is moved, a crack in the surface could appear as shown in figure 4.6 because one edge between two vertices is no longer coincident with the more deeply refined edge of three or more vertices.

Figure 4.6: A crack (the shaded triangle) appears between squares A, B and C because the marked vertex joining B and C should be on A's edge. THRED avoids this condition by disallowing edits of these vertices.

Because of this, there are in fact two subsets of S - S_{refine} and $S_{reshape}$. S_{refine} contains all the vertices that the user has selected, while $S_{reshape}$ eliminates those vertices that do not meet the edge-to-edge vertex matching restriction. If S_{refine} and $S_{reshape}$ are not the same, then $S_{refine} \cap S_{reshape}$ is outlined in yellow piping, and $S_{refine} - S_{reshape}$ is outlined in red piping. If there is no red piping then the entire yellow-piped region is both resizable and reshapable.

The polygons of yellow and red piping are determined using a 4-connected floodfill algorithm on the grid. A seed vertex is chosen from S, and the standard floodfill algorithm is followed until there are no connected vertices in S that are 4-connected to the seed. This 4-connected polygon is passed to a plane sweep algorithm that passes from North to South, and within that from West to East. At each vertex, the number of edges that connect to other members of the 4-connected component are counted, and used to determine what piping to draw to the neighbors.

The presence of uneditable vertices is the obvious disadvantage of this hierarchy of rectangles. The advantage of this scheme is that there is a naturally-preserved hierarchy of fine geometry that can ride on the back of coarse squares, so that global edits can be performed by moving a few coarsely-refined vertices.

4.5.2 Geometry of Control Point Selection

Given that the user desires a control point, this section describes how the user geometrically controls the right bat to pick the desired point.

THRED uses a probe selection technique similar to Liang's [41, 42] *spotlight*. A probe, represented by a narrow cylindrical shaft, is attached to the right cursor, and the user controls the position and orientation of the probe with the right bat. The distance from this probe is computed for each control point using a specialized distance metric called the *probe metric*, and the point that generates the smallest value is the current *hot spot*. To pick items in the scene, the user may both translate and rotate the right bat to point the cone at the desired items.

The probe has two visual presentations; one for selecting objects, and the other for selecting a single point on the surface of an object. For surface selection, an unambiguous and easily-

understood cursor point is needed. To achieve this, the probe metric is evaluated for all vertices, and those vertices with large metric values are eliminated from further consideration. Next, all surfaces adjacent to the remaining small-metric vertices are intersected with the probe axis. The intersection that is at the minimum Euclidean distance from the right cursor is the chosen surface point. This intersection point is highlighted by drawing a small sphere, and I draw the probe axis from the right cursor to the intersection point.

The hot control point is the vertex closest (in Euclidean distance) to the intersection point on the intersection triangle. The hot spot is highlighted by drawing an arrow from the intersection point to the hot spot. This highlight arrow lies in the surface of the triangle where the intersection occurs. The right image of figure 4.8 shows the probe and the hot spot arrow.

If no objects intersect the probe axis, the probe visually switches to object selection mode. To select objects, the probe axis is drawn to some arbitrary length, surrounded by a translucent cone which has a radial spread angle of 10 degrees. The probe is the cone axis, as shown in the left image of figure 4.8. A spotlight that has identical visual geometry to the translucent cone is attached to the cursor. As the user sweeps the probe about the scene, the objects that fall within the cone intersect visually with the cone wall, and are highlighted by the spotlight. Since vertex selection is being performed, an arrow is drawn from the right cursor to the nearest (using the probe metric) control point. When there is an intersection between the probe axis and a surface, the translucent cone and the spotlight are turned off, because both tend to obscure surface detail.

The non-Euclidean probe metric is designed to closely mimic the shape and behavior of the probe axis. Given a point, the smaller the angle between the probe axis and a line from the cone apex to the point, the smaller the metric. If two points have equal angular distances from the probe axis, the point that is closest in Euclidean distance to the cone apex yields the smaller metric. The probe metric also allows the selection of objects that are small and/or far away. Simply tracing a mathematical ray along the probe axis and picking the first intersecting object is not effective, because a high degree of cursor accuracy is required to pick small objects.

Given a point $P = (x, y, z)$, and given that α (measured in degrees) is the angle between the probe axis and P, then the probe metric from P to the origin is given by

$$ProbeMetric = |P|(1 + \alpha) ,$$

where $|P| = \sqrt{x^2 + y^2 + z^2}$ is the usual Euclidean distance.

To compute the probe metric from a cursor located at point C, with the probe axis directed along the normalized vector R, the vector V from the cursor to P must be generated:

$$V = P - C \qquad V_n = \frac{V}{|V|}$$

Since R and V_n are unit length,

$$\alpha = \cos^{-1}(R \cdot V_n)$$

and,

$$ProbeMetric = |V|(1 + \frac{180 \cos^{-1}(R \cdot V_n)}{\pi})$$

The conversion from radians to degrees serves to sharpen the outline of the isodistance curve, as shown in figure 4.7. Any suitably large scaling factor would also serve well. Figure 4.7 also shows the isodistance curve for the standard Euclidean metric $|V|$, and for Liang's ellipsoidal metric[1]:

$$K = \sin(cone\ angle)$$

$$EllipsoidalMetric = |V| \frac{\sqrt{1 + (K^2 - 1) * (R \cdot V_n)^2}}{K}$$

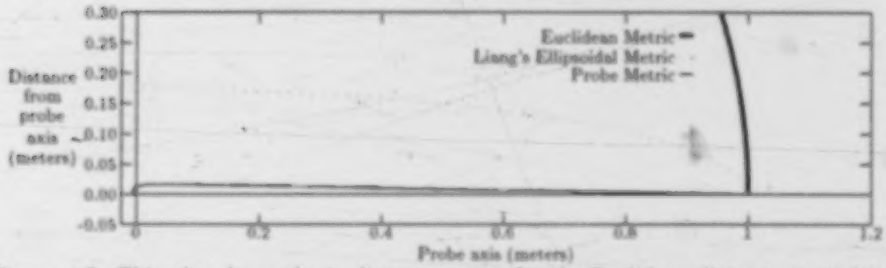


Figure 4.7: This plot shows the isodistance curves for the Euclidean distance (a circle), Liang's Ellipsoidal metric, and the Probe metric. The cursor is at the origin and extends along the horizontal axis. This plot is to scale.

There are some important differences between Liang's spotlight technique and the probe technique. The spotlight is only useful for selecting objects. It cannot be used without modification to select a point on the surface of an object, because of the ellipsoidal distance metric. This metric is rather difficult to control because when the probe axis is moderately close to parallel to the surface (less than 30 degrees), the roundness of the ellipsoid near the probe axis evaluates off-axis points on the surface to be closer than on-axis points. Also, the ellipsoidal metric must be cut off by the surface of the cone. If it is not cut off, the ellipsoidal metric selects surfaces that are (Euclidean) close to the cursor but which lie outside the cone. In sum, the ellipsoidal metric does not usefully approximate the shape of the probe axis unless the probe axis is almost perpendicular to the surface.

The probe metric does approximate the imagined needle-like shape of the probe axis, and it has the useful property of being equal to the Euclidean metric when evaluated on-axis. In fact, when the surface intersection test is turned off, the probe metric usually picks the vertex closest to the surface intersection point unless the probe axis is almost parallel to the surface (within 2 degrees).

A disadvantage with the probe metric is that in situations where the probe is perpendicular to the middle of a triangle, and is close to its surface, any vertices behind it that are close to the probe axis will give small probe metric values. Unless every surface is intersected with the probe axis, or an intersection-tracking scheme is used, a vertex that is close to the probe axis will end up being chosen as the hot spot. In this case, the user must point the probe at the desired vertex.

Volume Cursors

Kabbash and Buxton [35] make the salient point that picking a point with an area or volume cursor is easier than with a point cursor. An example of this is Zhai's [77] *silk cursor*, in which

[1] There is an error on page 502 of Liang's paper [41]. The correct formula is given here.

a translucent box is used to surround objects in the scene. However, because the user cannot conveniently control its size, the silk cursor is only good for selecting widely separated points or objects.

Liang's spotlight cone does not have this problem because the cone has a linearly increasing cross-section, and the user controls its position and orientation. To pick one of two small adjacent objects, the user can move the cone close to the desired object (reducing its cross-section), and reorient the cone so that the ellipsoidal metric unambiguously picks one object. However, as previously discussed, the ellipsoidal metric must be cut off by the cone, or it will give false positives at nearby points outside the cone.

The probe metric solves the false positive problem by using an isodistance shape that closely approximates a line segment. Unlike the Ellipsoidal metric, the probe metric is unidirectional, extending only slightly "behind" the cursor. Thus, while the probe may be thought of as a volume cursor, the shape of the probe metric allows us to dispense with an explicit point-in-volume test. The metric allows us to generate a partial order of all the vertices in the scene. Vertices below some threshold are used as the most likely to generate a surface intersection.

Previously, a scheme similar to Bier's [5] was used, in which a line is constructed from the eyepoint through the cursor into the scene. A cone whose axis is this line and which has a 10 degree apex spread was used to initially reject control points. All points that lie within the cone were then sorted by depth, and the closest control point is the current hot spot for the cursor. This technique uses only the position of the bat to select, since a line is traced from the fixed eyepoint through the bat's location.

This scheme is functional, but it has the drawback that the user can only do selection by translating the right cursor. To select widely-separated points, for example when the entire surface is to be selected, the user's arm must therefore translate a fairly large distance. With the probe, the user can simply reorient the right bat with the fingertips to select widely-separated points. The probe also allows the user to select objects and surface features that are hidden behind other surfaces, which is impossible with Bier's scheme.

4.5.3 Probe Orientation

All of the previous section assumes that the probe axis is aligned to some arbitrary vector through the right bat. Mathematically, it makes no difference what vector relative to the right bat's coordinate system is used as the probe axis, but from a human factors standpoint, it matters a great deal. Liang used the X axis of the bat (see figure 4.2) as the spotlight cone axis because this is the axis along which the wire connects to the Polhemus sensor.

With buttons attached to the Polhemus sensor, using the X axis makes it difficult to comfortably point the probe at the target while pressing a button. In particular, vertices on the lower half of the screen were hard to pick because the natural posture[2] for the right bat is with the X axis pointing straight up (figure 4.2). The Z axis was tried next, because in the usual posture, the Z axis of the right bat points towards the screen. The more central orientation of the Z axis gives the right hand

[2] See section 4.7 for a discussion of this issue.

more equal access to all of the vertices.

The Z axis is still not comfortable, so a vector that would closely match the line that a pen would take in the standard grip was tried. The pen concept was based on the idea that since people are used to drawing with a pen, the usual orientation of the pen in hand would be easily remembered. Unfortunately, this is not the case, and the salient issue here isn't spatial memory but spatial accessibility.

The ideal probe orientation is perpendicular to the screen when the hand is in a relaxed pose. This orientation can be generated by simply tracing a ray from the virtual eyepoint through the cursor when the user is in a relaxed pose. The line is transformed into the coordinate system of the right bat, and a vector parallel to that line is attached to the bat's coordinate system and is used as the new probe axis. Once the probe realignment command is issued, the geometric calculations to determine the new probe orientation are performed, and the user can now position and orient the probe using the new probe direction.

The difference between the probe technique and Bier's technique is that Bier's technique is always tracing a line from eye through cursor and doing a depth sort along this line. With the probe, the selection vector emanates from the cursor, and the position and orientation of this vector is controlled by the bat. Bier's depth sort is replaced with the probe metric evaluation.

This reduces the amount of fatigue that users experience, since users do not have to twist their right arms into funny shapes in order to reach moderately distant vertices. The equivalent procedure in a mouse-based interface is to pick up the mouse from its current location and move it to a more comfortable position. When the mouse is put down, it can once again be used to move the cursor.

Another benefit of this is that in comfortable orientations, the user can successfully draw on the surface, even on curved surfaces. With a little practice, the user can trace along a contour line on the surface.

4.6 Left Hand Operations

The left hand has four tasks to perform, all related to context setting in some way:

- Set the current constraint mode.
- Set the current selectable refinement level.
- Set the current reshape shape and invoke miscellaneous commands.
- Manipulate the position and orientation of the entire scene.

Scene orientation is a toggle controlled by left button 3. Pressing button 3 attaches the workpiece to the left cursor, and pressing it again leaves the workpiece in place. This clutching mechanism does not use continuous button depression because the user spends a large amount of time moving the workpiece around, and continuous button pressure would be a significant burden.

This mode is similar to 3-Draw [53] and WIM and related systems [30, 66], with the exception that in THRED, it can be turned off. Moving the model while doing control point editing sometimes

adds a certain amount of inaccuracy due to tracker noise, so users generally turn it off while refining the design.

4.6.1 Constraint Mode

When manipulating the control points, there are usually two phases: gross approximation followed by many steps of refinement. Initially the user wants to quickly place control points in approximately the right place, and then at some later point the user will refine these points to greater and greater precision.

Users also generally find precise placement of objects much easier to control if they are controlling only one degree of freedom at a time. Users also do not want to wreck previous refinements by accidental movements caused by refinements in another degree of freedom. Liang [41, 42] points out that a substantial fraction of the operations that users perform in JDCAD are axis-constrained operations, and my experience in using JDCAD bears this out.

Therefore, both constrained and unconstrained operations are available in THRED, with the left hand providing both the type of constraint and the geometry of the constraint. Clicking left button 2 moves from the current constraint mode to the next in a three-step cycle of *no* constraints, *line* constraints, and *plane* constraints.

Initially, the right cursor can manipulate vertices with all 3 positional degrees of freedom. When left button 2 is clicked, the system moves into line-constraint mode, in which right-handed reshaping operations are constrained along one of the three major axes. Both cursors change their shape to reflect line constraint mode, with the left cursor changing to a short pole through a square, and the right cursor adding a short pole that snaps to the current constraint axis (figure 4.8). The left cursor does not snap but remains tied to the left bat, because the canonical axis closest to the bat's Z vector (the pole axis in the left cursor) determines the constraint axis.

Figure 4.8: Left: Line constraint mode is active. Selected vertices can only move vertically. The right cursor shows the selection cone with its central probe axis, and an arrow pointing from the cursor to the nearest vertex.
Right: Plane constraint mode is active. Selected vertices may move in the vertical plane shown by the left cursor. The probe on the right cursor intersects the surface, and the hot spot is indicated by a small arrow from the intersection point.

Thus while the left bat is geometrically stating the constraint axis, the right axis can be making a selection and starting the reshape operation. In order to avoid any inadvertent changes in snap axis, the constraint axis remains fixed while the reshape operation takes place, so the left hand does

not need to hold a steady orientation during reshaping.

Similarly, in plane constraint mode the canonical plane of motion is stated continuously and directly by the orientation of the left hand, and remains constant while the right hand is doing a reshape. In this case, the right cursor shows a small square that is snapped to the constraint plane, and the left cursor shows a much larger snapped plane as shown in figure 4.8. The left cursor also has a small square rotating freely with the left bat, to help the user get to the desired orientation.

4.6.2 Selecting Refinement Level

All of the operations discussed above take place when the operating cursor is within the main window. However, the left cursor can select the current refinement level when it is moved above the top edge of the main window into the refinement level window.

The geometric space of these two windows is contiguous, and is kept consistent between window resizes so that when the left cursor moves above the main window's top edge, it appears in the same place in the refinement level window.

Each level is represented by a square triangulated in the same NorthWest-SouthEast pattern as the edited surface, and is drawn using the same level-based colors. The squares are screen-aligned and are arranged from left to right joined edge to edge. The number of levels evenly subdivide the window horizontally, so if there are 3 levels, the level 0 rectangle takes the left third, the level 1 rectangle takes the middle third, and the level 2 rectangle takes the right third. The currently selected levels are outlined with yellow piping.

When the left cursor is moved into the refinement level window, it is projected onto the surface of the refinement squares, and converted into a 1D slider. To find which level is being intersected by the left cursor, just the horizontal position of the cursor is used and range checks are performed for the boundary between each square. Thus, precise location of the left cursor within the small window is not necessary, because the user just has to move into the refinement window. This low-precision selection mechanism helps in enlarging the targets so that they can be hit quickly [21].

The cursor is an arrowhead pointing over the current location located just in front of the squares. To select the level under the cursor, the user just clicks the left button 2 once. To select more than one level, the user holds down button 2 at one end of the selected set and drags the cursor to the other end. Double clicking the mouse button toggles between the current selection and selecting everything, so the user can quickly step in and out of select-everything mode with a double click in the level selection window.

4.6.3 Selecting Reshape Operators: The Ring Menu

Since the user holds a bat in each hand, it is inconvenient to drop the right bat in order to perform menu selection with the mouse. A means of selecting among a group of choices using the bat is needed. The first attempt at this was to use and extend Liang's Ring menu, (figure 4.9), which is a popup menu controlled by the orientation of the left bat. The menu items are placed on the circumference of a circle at a fixed orientation with respect to the bat. The translucent belts serve to visually disambiguate the items in the front half of the ring from the back items, and the

gap in the front belt supplies a selection zone. The belts are maintained on the Y-Z plane of the bat with the selection gap always pointed at the user, so as the user rotates the bat about the X axis, the item to be selected appears within the gap and is highlighted.

The standard selection syntax for a popup menu is

$$Press - Drag - Release.$$

That is, the user presses the button, drags the cursor to the desired item, and releases the button.

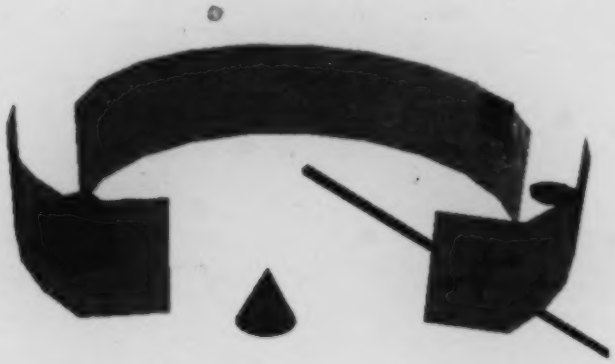


Figure 4.9: The ring menu. Cone reshaping is currently selected. Tandem move is half behind the translucent band on the left, and scale, scale to median plane and adopt parent are on the right around the circumference.

To extend the ring menu to use a 6 DOF tracker with buttons and the standard selection syntax, there are five main difficulties:

1. Using a bat button to pop up the menu limits the range of motion to about 3/4 of a revolution. The ring menu was designed to work without buttons, so that the user could twirl the Polhemus sensor about the X axis using its wire (see figure 4.2). When a bat button is used, the user must rigidly grip the bat through the entire menu selection process, which means that the user cannot extend the bat's orientation envelope by unclutching and clutching it. While the button is pressed, the bat has approximately the same orientation envelope as the wrist.

2. The translucent belts tend to obscure the items in the menu. Users typically reorient the Y-Z plane of the ring so that they can examine what choices are available.

3. Text items are visually confusing. The distribution in depth of the items and the translucency of the belts make text items almost guaranteed to be hard to read.

4. Users must sequentially pass through items in order to reach the desired item. Standard rectangular menus have this difficulty also.

5. There is no way to cancel menu selection. Once the menu has popped up, the user must choose one of the options. This could be fixed by allocating a blank area on the ring as a no-op item.

6. The ring menu is not conveniently hierarchical. A child menu cannot simply pop up when its parent pointer is highlighted, because the user may be on the way to another item. Some spatial means must be chosen to indicate that a child is desired, but none is readily at hand. Users typically use the two remaining degrees of orientational freedom to examine what choices are available, and they reposition the menu so that they can see the items more clearly. For this reason, the ring menu is limited to about 20 items.

Ring Discussion

An alternative that is used by Motif is to add the selection syntax

$$Click - Drag - Click,$$

where click means *press-and-release*. In Motif pull-down menus, the button release must occur on the parent item in order for the pull-down menu to remain available. This allows the *Press-Drag-Release* syntax to be available also. If both syntaxes are to be made available in the ring menu, the item that appears in the selection gap on popup must have a similar function. Otherwise, this start-up item could only be selected by clicking twice, because no dragging is needed to reach it. Because no dragging is needed, there's no way to tell if this is *Click-Drag-Click* or *Press-Drag-Release*. All others on the ring could be selected by both syntaxes.

If *Click-Drag-Click* is used, the character of problem number 1 changes. Popping up the ring menu with a button click allows the user to release the button while dragging the cursor. The orientation envelope is thus once again the full range, because the user can unclutch the bat and reclutch it at the destination. However, when the destination is reached, the user must click the same button, which may well be out of reach.

Since the location of bat's buttons is fixed, the user will grip the bat in a posture where the fingers or thumb most readily press the buttons. The user will also adopt a posture where the hand is most relaxed. This is the situation when the first click pops up the ring menu. If the target is within the 3/4 revolution envelope of the wrist, then unclutching is not necessary, and the button will be under the user's finger when the target is reached. If the bat is out of the wrist envelope, the bat may need to be unclutched, which will move the buttons to a less convenient location, making them more difficult to reach, much less press. In this situation, the user must fumble to maintain the bat in the target orientation while attempting to click the button. In the bat described in section 4.2, button 1 is the only sensible choice, because the grip required to twirl the bat about the cable will supply the support needed for index finger pressure on the button.

This syntax also allows the use of a second button as a cancel button, but this adds complexity to the user's understanding of the syntax, because it puts an additional function on a button whose main purpose is something else. With respect to the previous paragraph, the cancel button must be button 2 or 3 which means that when the bat is unclutched, the cancel button is somewhat harder to hit. It also requires at least 2 buttons on whatever bat is being used, and makes it possible to implement only 2 menus on a 3-button bat or 1 menu on a 2-button bat.

Ring hierarchies are possible if hierarchical parents are speed sensitive, so that when the user stops on a parent item, the child pops up. The problem with this is that when the user is unfamiliar

with the menu, there is a tendency to hold the menu still, which may accidentally pop up a child. This is especially important at the extremes of the user's orientation envelope, and at the starting item. Also, since the belts obscure the items, the user will periodically turn and hold the menu to look at other objects, meaning that accidental popups may occur anywhere.

Another cause of false popups occurs in overshoot situations, which are especially common in interfaces with low update rates. When the user overshoots the desired target and reverses direction to return to it, if the change in direction occurs over a parent, the direction change will be read as a slowdown and the child will accidentally pop up. Again the cancellation issue comes into play if the user didn't select the correct parent.

A more subtle difficulty arises when the user has indeed selected the desired child. Parent items that are at an extreme of the orientation envelope will pop up children that can only be selected by rotating the hand in the opposite direction. That is, objects which are spatially nearby may in fact require almost a full revolution in the opposite direction to reach. If the *Click-Drag-Click* syntax is used, then the user must be careful when unclutching the bat to avoid accidental popup at the unclutching point.

It seems then that the slowdown technique requires that parents be near the center, because the extremes cause problems with the child menu geometry and overshoot. However, putting parent items near (but not at) the center increases the probability that they will be passed over on the way to something else. The user must therefore move fast over these items, which in turn increases the probability of overshoot.

In summary, the possible techniques available to "fix up" the ring menu all add a significant burden on the user's manual dexterity. The *Click-Drag-Click* syntax expands the menu's range but demands more grip management from the user. The slowdown popup of child menus requires the user to keep the menu in motion until the desired item is selected. The cancel button adds to the user's cognitive burden, and adds to the dexterity burden when the bat is unclutched.

4.6.4 The Sundial Menu

To avoid these problems, THRED uses the sundial menu (figure 4.10), which is a popup menu controlled by the orientation of the left bat. The sundial menu builds on Liang's [41, 42] observation that interaction techniques can be controlled by orientation. The menu choices are arrayed on the circular plate of the sundial, each on its own pie-shaped sector. The desired item is picked by pivoting the *shadow stick* about its base so that the stick's endpoint lies visually in front of the sector. The user controls this shadow stick pivoting by manipulating the bat's orientation. The base of the shadow stick is located at the center of the plate, and when the menu first pops up, the stick is aligned with the line of sight, pointing directly at the user. Around the base of the stick is an inner circle which the user can point the stick at to indicate *no selection*. The position of the sundial is fixed rigidly to the left bat, and the dial lies parallel to the projection plane. I implemented the *Press-Drag-Release* selection syntax, although the *Click-Drag-Click* syntax would work with equal success. Also, the shadow is not actually drawn in the current implementation.

The sundial is a 3D version of the pie menu [12] which encodes the position of the 2D cursor

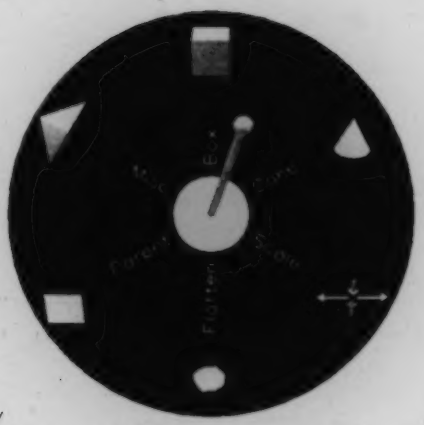


Figure 4.10: The sundial menu. Tandem reshaping is currently selected. Clockwise from tandem is cone, scale, scale to median plane, adopt parent, and a child menu.

by projecting the 3D position of the endpoint of the shadow stick onto the 2D sundial plate. To implement this, I trace a ray from the eyepoint through the endpoint of the stick, and intersect it with the sundial plate. Under parallel projection, this can be simplified by simply taking the X and Y components of the endpoint's offset from the dial center, but this results in visual inaccuracy under perspective projection. Similarly, the realignment of the shadow stick along the line of sight at popup time is needed so that the stick starts at a neutral orientation. Realignment also has the advantage of allowing the user to have the bat at any orientation when the menu pops up.

I set the length of the shadow stick equal to 1.2 times the radius of the dial, which allows the user to cast a shadow on the outer circumference of the dial with a deflection from center of 56 degrees. A deflection between 56 and 90 degrees is of course still valid, since the nearest sector is highlighted. A longer stick allows smaller deflection angles to reach the circumference, but makes the central *no selection* circle slightly harder to hit. The advantage of a longer shadow stick is that it feels more responsive when ballistic rotations are made, and it reduces the amount of rotation necessary to move the endpoint into the desired item. Currently, the inner circle is 0.2 times the radius of the dial, which means that a stick of length 1.2 must be deflected within 9.6 degrees of center to select nothing. I feel that this is a good trade-off between the fine control needed to move to a small target (the center) and the speed needed to move quickly to a large target. Controlling shadow stick size is not really a good means of manipulating control-to-display (C/D) ratio, however. More direct means should be used, such as multiplying the rotational offset from center by some factor.

Hierarchical Sundial Menus

Hierarchical menus are supported using the sundial menu, as shown in figure 4.11. The purpose of the hierarchy is of course to allow for a larger collection of choices than can be conveniently and legibly placed on a single menu. As figure 4.10 shows, a child menu is represented by an outward-pointing arrowhead.

The user moves the shadow stick into the parent sector to pop up the child. As a visual reminder

of where the parent is located, the child pops up with its center located over the arrowhead of the parent sector. The child is popped up only if the rotational velocity of the bat drops below a certain threshold while in the parent sector. This ensures that the shadow stick is not in motion when the child appears.

To pick a top-level item, users typically make a ballistic rotation towards the desired item and then release the button. Users expect similar behavior when a child menu is to pop up, and they also expect the shadow stick to appear stable in the child's center when the child appears. Without the velocity check, the child would pop up immediately on parent entry, and the shadow stick would continue to move as the user completes the parent-selection rotation. Because the shadow stick is realigned to the line of sight at child popup, continued shadow stick motion causes the user to inadvertently select one of the child items. The time it takes for the user to recognize that the child has popped up and to accordingly stop rotating the bat is enough to allow the shadow stick to turn another 10 to 20 degrees. This is exacerbated by the lag in the tracking system (about 100ms), which increases the time it takes to stop the shadow stick. The maximum velocity check also allows the user to quickly pass over a parent menu item without popping up its child.

The difference between this slowdown technique and the slowdown technique proposed for the ring menu is that all sundial selections can occur from a neutral location without passing through other items. By contrast, ring selections must by definition pass over intervening items to reach the destination, so if the intervening items are parents, then the user must move fast. The user has no spatial recourse with the ring menu.

To return to the parent when a child has been incorrectly entered, a user may simply move the shadow stick to the no-selection circle and release the button. I experimented with using an inner *backup zone* inside the no-selection circle, which pops down the child menu and recenters the shadow stick on its parent when entered. The purpose of this zone is to satisfy the user's desire to avoid "starting from scratch" when they have accidentally selected the wrong child menu. The inner backup zone is not terribly effective, since it is hard to hit, and in conditions where there are three or more levels of menus, entering the backup zone of level 3 will cause a backup to level 2, and the recentering will cause a backup to level 1. Without recentering, the shadow stick reappears once again on the parent sector, and since the bat has low rotational velocity, the same child once again pops up.

More consistent with the idea of backing up to the parent menu is to use the volume of space behind the child as the backup zone. To get to this zone, the user turns the shadow stick so that it is behind the child menu (90 degrees of deflection or more), as if trying to touch the parent menu. Entering the zone immediately causes the child to pop down. I do not recenter the shadow stick because the user's hand is usually in an awkward orientation, so the user recenters by moving back to a comfortable orientation. This scheme has a similar problem of cascading backups if the user has turned the shadow stick too far behind, but this does not always happen. The advantage of this scheme is that backing up can be accomplished with a flick of the wrist, since the backup zone is large (the full hemisphere) and therefore easy to hit.

Figure 4.11: A hierarchical sundial menu. The child menu of the previous figure has been selected, and the probe orientation command ("Spot Align") described in section 4.3 is currently selected in the child menu.

Sundial Menu Discussion

Deering's 3D *fade-up* menus [17] have a circular layout similar to the sundial. The items of a fade-up menu are laid out in concentric circles, with more frequently used items on the inner rings. When the user presses a bat button, the menu pops up centered around and slightly behind the cursor location, and stays fixed while the button is pressed. Selecting an item is accomplished by translating the cursor to the desired item so that the cursor is within the item's capture volume. While the cursor point is within a capture volume, the menu item highlights, and the user may release the button to activate the item. Selecting nothing is simply a matter of releasing the button while the cursor is intersecting none of the menu items.

Menu hierarchies are supported. When the cursor falls within a parent item's capture volume, the parent menu is moved back, and the child menu appears centered at the current cursor location. Backing up to the parent is done by moving out of the child menu's large enclosing volume. Deering reports that users need some practice to get used to manipulating fade-up menus, implying that the capture volumes are somewhat small. By contrast, the sundial menu has a somewhat smaller no-pick area and somewhat larger menu zones. These two techniques are not directly comparable, because the sundial uses orientation and the fade-up menu uses position to perform selection.

The sundial menu shares some of the advantages of pie menus [12], such as the ability to pick any item with a short cursor movement, and being structurally easy to remember. The short strokes are possible because the sundial uses two degrees of freedom to select a menu item instead of just one. Also, the targets are comparatively large, and so are easy to hit [21]. As a result, pie menus give shorter selection times and lower error rates than linear menus [12].

The disadvantages of ordinary 2D pie menus are as follows:

- When horizontal text is used, pie menus get rather large because the text for each item must fit within a pie sector. The effect is most severe for the top and bottom pie slices [39]. This large size tends to obscure a large part of the screen.

- Diagonal text may be used to economize on menu size, but it is harder to read and hard to generate.

- Pie menus popped up near any edge of the screen must be warped to a more central location so that the user can see all the items. Hierarchical pie menus run into this problem at every child popup. Hierarchical linear menus only have edge collisions near the bottom or the right edges of the screen.

- There is an upper limit to the number of elements that can be put in a single pie menu. Linear menus with scroll arrows have a much higher limit to the number of elements available.

- Circular objects are not easy to generate in most window systems.

The sundial menu shares the pie menus' disadvantages of limited item count and harder-to-read diagonal text. Because I am using the vector-based Hershey fonts, text at any orientation is easy to generate. Diagonal text helps to reduce the size of the sundial menu, so that it does not obscure too much of the screen. However, obscuration is less of a problem in THRED than in 2D systems or in one-cursor systems for two reasons:

- The menu's position can be changed during the interaction by simply translating the left bat.

- The user is usually concentrating on what the *right* cursor is doing. The left cursor is usually on the left side of the screen, and does not perform tasks that require the user to move it into the central area.

If the menu is obscuring something important, the user can move it out of the way without affecting the menu selection. This operation is entirely natural, and does not require the user to issue a command. The menu does not occupy a fixed amount of screen space, because as the user moves the bat farther from the eyepoint, the menu appears smaller due to perspective. For the same reason, popping up the sundial near the screen edge is not a problem. The user can move the sundial away from the edge to expose the obscured items if necessary. This natural translation of the menu allows for the user to continuously customize the location and size of the menu without any cognitive effort, and without having to restart the menu selection at a more convenient location.

A unique feature of the sundial and ring menus is that the size of the menu target zones remain constant, regardless of their size on the display. That is, because they are 3D artifacts using 3D input, the manual effort required to hit the target is constant, while the visual target size depends on how close the menu is to the eyepoint. As users learn the menu layout, they can take advantage of this fact, and pop the menu up at locations that result in smaller visual sizes. Experienced users can further optimize sundial menu selection by performing the ballistic rotations without actually looking at the menu.

A similar no-look strategy is used with marking menus [39, 40], in which the experienced user just makes the 2D strokes to select a menu item. The stroke path is drawn on the screen, so as the user descends the menu hierarchy, a zigzag path appears. A pie menu pops up after a small delay if the user holds the mouse still, so beginners must wait a little to find the next menu. Although sundial menus pop up immediately, ballistic motion is still available. With marking menus, the user can "back up" the ink trail to undo an erroneous selection. This is similar to using the central region of the sundial as the backup region, which has the problems outlined above.

The sundial menu has none of the difficulties of the ring menu:

- The range of orientation change is small, typically 30 degrees.
- All items are visible at once.
- Any item can be reached directly, without passing through other sectors.
- The sundial is hierarchical.
- Menu selection can be canceled by selecting the inner no-pick region.

4.7 Device Ergonomics

Using a stylus instead of one of the bats for the right hand is an attractive possibility, since Polhemus sells one, and the stylus might be used as a pen. I did not consider using a stylus for three reasons:

- The Polhemus stylus has only one button.
- The cable hangs off the end of the stylus, and therefore there is a tendency for the stylus to point straight up, to minimize the work needed to hold it.
- Because the stylus shape is mimiced by the probe, the probe therefore cannot be reoriented relative to the stylus as explained in section 4.5.3.

Thus, the stylus and the probe must always be parallel, or the whole point of visual mimicry is defeated. The stylus must therefore be perpendicular to the screen to put the probe in a centered orientation. A simple experiment with taping one of the bats to a pencil indicates that this orientation generates a certain amount of wrist fatigue, because the most relaxed orientation for the stylus is pointing up 45 degrees. Selecting items on the bottom half of the screen is a chore that requires either a large amount of wrist flexion, or that the user lift the right elbow off the arm of the chair. Thus the ergonomic consequences of using a stylus reduce its appeal somewhat.

4.7.1 Device Fusion

Most commands are entered using the bats, but when the need arises, the user can quickly put down one or both bats and move the mouse or type at the keyboard. The mouse is only used to activate the pull down menu, and to move windows around and access other applications. The keyboard is used to type filenames, and enter numbers for vertex coordinates. Also, single keystrokes are sometimes used to invoke commands that are in the prototype stage. However, unlike text entry, pressing a single key does not require that the user actually drop a bat. Instead, the user can press a single key with an outstretched finger while still holding on to the bat. Typically the hand that is not performing a spatial operation at the time of the command is used to hit the key.

As might be expected, the 6 bat buttons are prime real estate, so commands which are executed infrequently such as *scale scene* are put in the left sundial menu, as shown in figure 4.11.

The bats are small, so putting them down does not require that the user find large amount of desk space to put them. By contrast, the left clipboard in 3-Draw [53] requires about the area of a steno pad to lay on the table, which means that this amount of area must be free when the left hand needs a rest. Similarly, 6 DOF mice require a mouse-size area to put them when they are not in use, and in the THRED prototype, there are would be three mice beside the console, which is confusing.

There is some potential for confusion between which is the left bat and which is the right, and while visual labels are effective, I also find that a tactile difference between the two bats can be quite good for telling the user which bats is being manipulated. The left bat is also used on a DataGlove, and has a patch of velcro on it, while the right bat does not. When the user picks up both bats, the presence or absence of velcro tells immediately which bat is which. Figure 4.12 shows the author using THRED.

4.8 Sample Surface

Figure 4.1 contains a screen shot of a surface designed using THRED. It is a rough approximation of the Standish Face of the Sunshine Village Ski resort at Banff, Alberta. The lift line goes from the bottom left to the top left, and skiers with moderate skill (i.e. me) typically ski into one of the two bowls to the top center and the top center right. Experts ski down the face in the top left. This model contains 428 polygons, and was created in 18 minutes from my personal memory of the place.

Figure 4.12: The author using THRED. The user is in a relaxed position with the weight of each forearm balanced on each elbow.

Chapter 5

Experiment

This chapter describes an experiment that evaluates various aspects of THRED and compares its effectiveness with respect to other user interface styles.

There are several questions which naturally arise when a new interface style is introduced, as follows:

- Is the new interface style effective?
- Can the user perform tasks faster with the new style?
- Does a user's manual skill affect performance?
- Do users like the new style?
- Does the new style have any human factors concerns?
- How long does it take users to learn the new style?

5.1 Experimental Approach

The main thrust of this chapter is to answer these questions by experimental means.
THRED has three major parts:

- The interface.
- The surface data structure.
- The surface operations.

Each of these parts can be changed without significantly altering the structure or the interface to the other parts, which means that different interfaces to the same data structure and geometric operations can be validly compared. With THRED, interface styles can be compared alone, because they are based on the same underlying operations, and are performing the same fundamental task.

Comparisons of user interface styles and techniques sometimes lack a stable basis of comparison because the interfaces being compared are interfaces to two different programs, with different functionality. Because of this difficulty, interface studies often concentrate on small, easily-described tasks, such as picking, menu selection, and so on. The drawback with this concentration on minutia is that one may miss the big picture. That is, each of the little interface parts may be well characterized, but the application that integrates them may not work well at all. Conversely, techniques that work only moderately well in isolation may fit together more effectively because the drawbacks of each of the techniques are hidden by the actions of the other techniques.

Therefore, my experimental approach was to build three different user interfaces to THRED's underlying geometric operations, and have experimental subjects perform the same simple task on each interface over multiple trials. In conducting this experiment, it was important to establish a baseline of measure so that both timings and ratings can be based in a reasonably well-understood context. Therefore, each user's subjective impressions and ratings of comfort were measured using a repeated series of questionnaires that tracked the user's response over time.

5.2 Competitive Interfaces

With this study, three interfaces to the underlying database and operations available in THRED were built. The first interface is of course the two-handed interface, with a 6 DOF tracker for each hand. The second interface uses only one 6 DOF tracker, which leaves the other hand free to enter commands at the keyboard or to use the mouse. The third interface uses the traditional mouse and keyboard, in which the mouse does geometric manipulations and menu selection. With these three interfaces, the comparison has a stable basis, because the operations and the data structures are the same for each. Section 5.2.1 describes the mouse-based interface, and section 5.2.3 describes the one-handed interface.

5.2.1 Mouse-Based Interface

The mouse-based interface is a highly-refined commercial product currently in its seventh major release called the Alias Modeler, available from Alias Research in Toronto [1]. The modeler is part of a much larger general modeling and rendering system called Alias Studio. It is used by all of the major automobile companies worldwide, and was used to create the dinosaurs in the movie *Jurassic Park*.

The Alias interface presents the standard three orthographic views and one perspective view, in which the mouse can perform spatial operations. Below the 4 views is a menu bar that contains 16 *pull-up* menus that operate similar to the pull-down menus of the Macintosh: When the user presses the left mouse button in a parent menu item in the menu bar, the child submenu appears in the screen area above the parent menu item. As the mouse cursor moves into the submenu, the item under the cursor is highlighted. When the mouse button is released, the highlighted menu item, if any, is activated.

The Alias Modeler user interface is specially optimized to minimize the number of menu selections that the user must perform. More frequently used commands are placed in pull-up menu slots nearest

the menu bar, so that they are easiest to hit. The parent items in the menu bar are quite large, and thus also easy to hit. A few heavily-used commands like vertex selection and vertex motion are modes, so that the user does not have to continually select menu items to continue these operations.

Each pull-up submenu has "sticky" items. That is, the most recent submenu item is remembered, so the user can just press and release the mouse button on the parent item to again activate this sticky submenu item. Of course, the user may select another child item as necessary. The name of the remembered sticky item is displayed in a box below the parent item. Infrequently-used commands in a submenu are not sticky, because remembering them would overwrite the more frequently used commands.

Thus, a careful analysis of typical use has allowed Alias Research to create a highly effective 2D user interface. Typically, the user needs to switch rapidly between two or three modes, and this can be accomplished by pressing and releasing two or three easy-to-hit parent items.

A clone of this interface was created that exactly duplicates the interface functionality of the Alias modeler, including the commands that are modes, the menu items that are or are not sticky, and the window navigation operations. By *window navigation operations*, I mean the commands that change a window's view of the scene. That is, *navigation* is equivalent to *locomotion*. *Wayfinding*, a common alternative meaning for navigation, is not implied. Only operations that were appropriate to THRED were implemented, so for example, the menu items that deal with curves were not included in the clone. The purpose was to create the best possible competition for the other two interfaces, and thus perform the fairest possible comparison.

5.2.2 Alias Clone Details

The screen layout of the Alias clone is similar but not identical to Alias, which occupies an X window that takes up the entire screen. The majority of this window is the work area, which contains either one or four subwindows that display various views of the geometry being edited. Below this is a menu bar that has 16 menu items distributed over the entire width of the window. Below that is a single command and status line, and below that is 16 text labels that correspond to the menu buttons and display each menu's most recently selected item. The Alias clone does not have the command line, and the work area is slightly smaller than Alias so that it matches the size of the work area in THRED.

The following three sections describe the details of the various elements of the Alias clone interface. The section titled *Window Navigation* describes how the user adjusts the viewing parameters of one of the four views of the object being designed. The section titled *Vertex Selection and Reshaping* describes how the user manipulates the mouse to select and reshape the surface. The section titled *Menu Operations* describes the remaining operations and modes.

Window Navigation

Listed from left-to-right, top-to-bottom, the four views presented by the Alias clone are *Top*, *Persp*, *Front* and *Right*. Each view has a title bar with the name of the window on the left, and a small collection of window navigation icons on the right.

Window visibility is controlled by the *Window* menu, and by the *Fullscreen* icon in each window.

The two possible states are that all 4 windows are visible and of equal size, tiled in a 2 by 2 layout, or only 1 window is visible and fills the full work area. If one window is full size, clicking a mouse button in the Fullscreen icon changes to the 2 by 2 layout. Clicking a mouse button in a window's Fullscreen icon makes it full size.

The *Window* menu has 5 items. Selecting *Window→All* invokes the 2 by 2 layout, while the remaining 4 items each name a window to be made full screen.

Window navigation is used to change the view of the scene in one of the four views of the object, and is divided into two classes: Orthographic and Perspective. The orthographic techniques are zoom and track, and these allow strictly 2D locomotion operations on the orthographic windows. The perspective techniques add various types of 3D scene rotation for the perspective window. Zoom and track are also available in the perspective window, but they act a little differently due to the perspective geometry.

Navigation is invoked by either selecting an item from the *Views* menu, or by selecting a view navigation icon in the title bar of the window to be moved in. When a menu item is selected, the Alias clone is put into a mode where the selected navigation technique is active until another menu-based mode is selected. To perform navigation in a window, the user must move the mouse cursor into the window, press a button, and drag the mouse. Navigation modes are used because users typically spend a nontrivial amount of time specifying a view before editing objects.

Title-bar navigation works by pressing the left mouse button in the icon, dragging the mouse in the desired direction, and releasing the button. A special feature of this operation is that the mouse cursor disappears while the button is pressed, and reappears again at the same spot over the icon when the button is released. This allows the user to drag the mouse great distances without running into window limits, and also allows the navigation icon to be hit repeatedly without having to move the mouse cursor to the icon target for each navigation command. Menu-based commands do not have this disappearing cursor feature.

Only Zoom, Track, and Tumble are available as title-bar navigation techniques. The navigation operations available in the title bar and in the View menu are as follows:

Command	Windows	Brief Description
Zoom	Orthographic	Dragging right expands the size of the scene. Dragging mouse left shrinks the scene.
Zoom	Perspective	Dragging mouse right *zooms in* – moves the viewer forward along the line of sight, closer to visible objects. Dragging left *zooms out*.
Track	Orthographic	Dragging the mouse moves all objects in the scene in the direction of mouse motion.
Track	Perspective	Dragging the mouse moves the objects in the scene in the direction of mouse motion. The scene is translated using a reference point 0.6 meters into the scene from the near clipping plane.
Tumble	Perspective	Horizontal mouse motion rotates the scene about the scene's Z axis. Vertical mouse motion rotates the scene about a line through the scene's origin parallel to the window's horizontal axis.

Another difference between title-bar navigation and menu-selected navigation is that title-bar operations control the rate at which the scene is moved. The farther the mouse is dragged, the faster the scene moves. Menu-selected navigation does not control rate, but instead controls position, allowing the user to directly grab the window contents and move the scene.

The navigation operations only available in the View menu are as follows:

Command	Windows	Brief Description
Dolly	Orthographic	Left button drags out a temporary wireframe box. This box indicates the portion of the scene that will fill the window at the end of the operation – thereby zooming in. Right button drags out a temporary wireframe box indicating where the the current window's contents will appear in the new scene – zooming out.
Dolly	Perspective	Left button slowly zooms in and tracks scene. Middle button tracks scene. Right button zooms out and tracks scene.
Yaw/Pitch	Perspective	Vertically dragging the mouse rotates the scene about a horizontal axis passing through the center of projection. Horizontal motion rotates the scene about a similar vertical axis.
Twist	Perspective	Horizontally dragging the mouse rotates the scene about the line of sight.
Previous	All	Clicking a mouse button returns the window to the view that existed before the previous navigation command.
Reset	All	Clicking a mouse button returns the window to the starting view.

Vertex Selection and Reshaping

The syntax of vertex selection is identical to THRED in that a set of selected vertices is maintained. Vertex selection operations change membership of the selected set. If a newly-selected vertex is not in the set, it is added. If it is in the set already, it is subtracted. Set membership is highlighted by yellow or red piping in all visible windows.

The *Pick* menu has two items: *Vertex* and *nothing*. *Pick→Vertex* is a sticky mode that allows the user to add or delete vertices from the selected set of vertices. *Pick→nothing* clears out the selected vertex set. It is neither sticky nor a mode, because it is usually followed immediately by vertex picking. However, it is placed nearest the menu bar, making it the easiest submenu item to hit.

The geometry of vertex selection in orthographic windows is to use a 50 pixel hot region for vertex selection. This is different from Alias, which uses a 4-pixel radius, because THRED continually highlights the nearest vertex to the probe with an arrow. Using Alias's smaller pick region would unfairly bias the user's picking performance in favor of THRED.

In the orthographic windows, picking is divided into two steps. The first step, determines the list of vertices that are closest to the cursor's XY position. The actual dimensions of the orthographic view are taken into account, so this uses two of the three position coordinates of the vertex. If there are no ties, then this is the selected vertex. Otherwise, the vertex with the most extreme value along the perpendicular axis to the window is chosen. Viewed from infinity, this is the closest vertex along the selection line in 3D.

In the perspective window, picking uses the probe metric. The center of projection is the origin of the probe, and the mouse cursor determines the point to pass through into the 3D scene. The cursor is assumed to be on the near clipping plane. No 3D cursors are drawn, nor is the selection cone or the highlight arrow. This is in keeping with the Alias style, which highlights only the objects in the selected set.

The *Xform* menu contains one of two reshape operators. To perform reshaping without changing orientation, the user selects *Xform→reshape*, which is a sticky mode. The user then drags the cursor in one of the three orthographic views. If the left button is used to drag, reshaping occurs along the 2 orthographic axes of that window. The middle mouse button moves the vertices along the horizontal axis of the window, and the right button allows vertical motion in the window. This corresponds to plane-constrained reshaping and line-constrained reshaping in THRED.

In the perspective window, reshaping occurs along one canonical axis at a time, with the X, Y and Z axes assigned to the left, middle, and right mouse buttons respectively. The user of course sees the surface deform in all four windows simultaneously.

The *Xform→rotate* menu item is also a sticky mode, and it allows rotation of the currently selected set of vertices about one of the three canonical axes. Again, the X, Y and Z axes are assigned to the left, middle, and right mouse buttons respectively, but this assignment is for all windows, not just the perspective window.

Menu Operations

This section lists the remaining operations not yet dealt with in the preceding sections. None of these menu items are modes in the sense of staying active after they have been selected. All items are sticky except where noted in the *Modify Squares* menu.

Menu	Brief Description
File	Contains New, Retrieve, Save, Save As, and Quit
Reshape Operators	Contains a list of the available reshape operators. This menu sets the current reshape mode, which is shown in the display box below the *Reshape Operators* menu bar item.
Modify Squares	Contains Add X Row, Add Y Row, and Refine. Only Refine is sticky, as the Add Row commands are comparatively infrequent.
Annotation Tools	Contains commands for toggling the various visual surface features.
Delete	Contains one item that deletes the vertices in currently selected set.
Level Display	This submenu lists the currently available refinement levels of the surface, and allows the user to select either all levels or a single selectable level by number. The display box below the *Level Display* menu bar item shows the currently selected level, so there is no separate level display window.

The *Annotation Tools* menu is identical in content to the menu of the same name in THRED. In addition, the *Annotation Tools→Tgl grid* command toggles both the XY plane grid in the perspective scene, and the screen-aligned grid in the 3 orthographic windows. The screen aligned grid is equal to the snap grid, so users may use this grid to verify exact alignment of vertices to snapgrid locations.

5.2.3 One-Handed Interface

The one-handed version of THRED moves all of the 3D geometric functionality to a single 6 DOF tracker in the right hand. There is only one cursor, controlled by the single 6 DOF tracker. The three buttons on the right tracker maintain THRED's functionality of Selection, 3D Reshaping, and 6 DOF Reshaping. The left tracker buttons no longer exist, so the left hand uses the 1, 2, and 3 keys on the keyboard to activate the commands that were previously assigned to the nose, dorsal, and tail buttons, respectively. That is, the 1 key toggles constraint mode, the 2 key pops up the pie menu on the right cursor, and the 3 key toggles reorientation of the 3D scene on the right tracker. If the left hand is holding the 6 DOF tracker, the user can invoke these commands using the 8, 9, and 0 keys, corresponding to 1, 2 and 3.

Because so much spatial functionality is assigned to the right tracker, some operations are either awkward or almost impossible. Like THRED, menu selection is possible while the 3D scene is being reoriented. This looks a little confusing, but the software allows it. Similarly, while selecting a menu item, the probe is turned off to minimize visual confusion. Aligning the probe while selecting the *Align* command from the menu is almost impossible, because the user's right hand will most likely be at a non-neutral orientation when the alignment command is activated. Thus, aligning the probe with the line of sight must be triggered by the "A" key.

Constrained reshaping operations are combined onto the right cursor. That is, the user must first align the constraint line or plane to the desired axis, then press the right dorsal or tail button to commence constrained reshaping. Once reshaping starts, the right hand need not maintain the alignment constraint.

5.3 Tasks

For this experiment, subjects were asked to perform multiple trials of the following two tasks. This section describes the tasks, and how to perform them, while section 5.7 describes the ordering of tasks for each subject.

Both tasks required the subject to create simple geometric shapes which could be easily described, recognized and created. Initially, the experimental plan called for the subjects to create a simple free-form surface that closely resembled a real object presented to them at the start of the trial. This offered the advantage of allowing the subject to verify the correctness of the synthetic object by visual comparison with the real object. However, it seemed that the subjects would spend so much time in verification that this task would essentially turn into a visual comparison task. It therefore seemed that the best object was one that could be described simply, and with measurements.

Task and interface learning is another issue of interest. An expert can perform the tasks described here in about 30 to 40 seconds. To get to this level, the subjects must practice the task a few times, which means that the task must be simple enough to complete in a reasonable time. Given that the power law of practice [13] is likely to apply, a task that experts complete in N seconds after years of exposure and abundant practice, is likely to take some subjects 10 times as long to initially complete. Therefore, a task that takes me 5 minutes to complete is not going to offer the subject much chance to do it more than once on one interface, let alone on two or three different interfaces.

Therefore subjects were asked to create simple geometric shapes. An easily-understood target shape minimizes the subject's cognitive burden, allowing greatest possible concentration on performing the task. The subject can quickly verify if the created object is correct, because it is easily understood. A simple shape also allows the subject to create it quickly, allowing many practice iterations.

5.3.1 Task 1

When the user selects the *New* command, the editor creates a single square quad 2 meters by 2 meters (figure 5.1). The task is to create a flat rectangular connected 3 by 3 array of 1 meter by 1 meter squares. This is the first task that any user would perform on the surface before continuing on to create a more complex shape.

To do this, the user must do the following:

1. Subdivide the surface in X two times. (the order of these first two operations does not matter).

2. Subdivide the surface in Y two times. The result is a 3 by 3 array of rectangles whose total size is 2 meters by 2 meters (right image of figure 5.1).

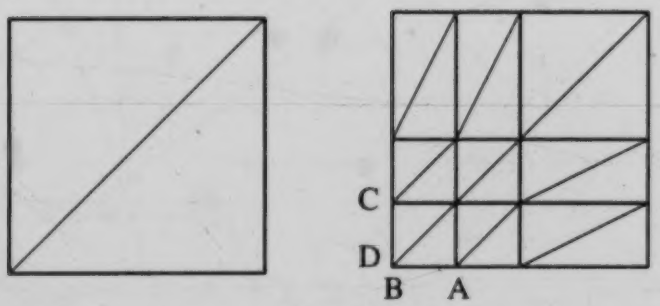


Figure 5.1:Left: The starting configuration of task 1.Right: After two X and two Y subdivisions.

3. Put a ruler on the 3 by 3 array of quads by selecting all quads, and then selecting *Annotations→Add Ruler*

4. Select vertex columns A and B and move them together to the left by 0.5 meters, giving the left image figure 5.2. The rulers on the surface can be used to verify that the width of the middle column of rectangles is 1 meter. Also, the ground plane grid can be used to verify exact snapping.

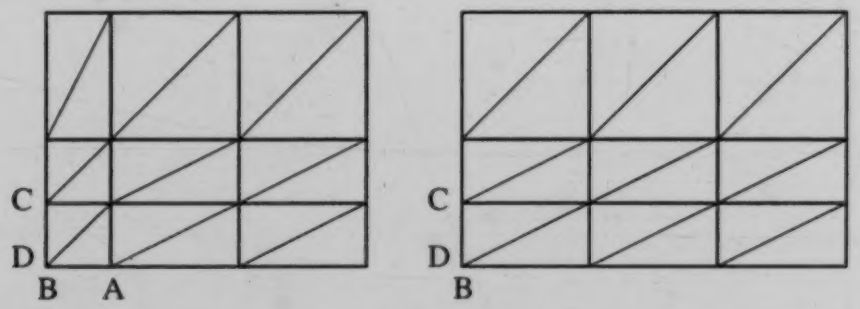


Figure 5.2:
Left: Task 1 after moving columns A and B together.
Right: Column B has been moved left. Next, vertex rows C and D must be moved down.

5. Select vertex column B (either by selecting column A to delete it from the selected set, or by selecting nothing and then selecting column B), and move it to the left by 0.5 meters, giving the right image of figure 5.2. The rulers on the surface can be used to verify the correct width.

6. Select vertex rows C and D and move them together down by 0.5 meters. Verify using the ruler and/or grid.

7. Select vertex row D and move it down by 0.5 meters, giving the final result in figure 5.3. Verify using the ruler and/or grid.

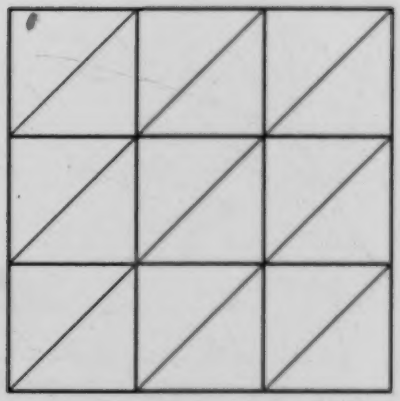


Figure 5.3: The completed task 1.

This task is best accomplished using a series of line constrained movements, so that the grid does not become crooked. The rulers and grid lines are used to measure the length and width of Subdivide the surface in Y two times.

5.3.2 Task 2

Task 2 is an enlarged version of task 1 that continues where task 1 leaves off. The goal is to create a 1 meter by 1 meter by 1 meter open box using the 2 meter by 2 meter square. The single square is first used to create the regular 3 by 3 grid of task 1, then the grid is used to create the open box. This is a simple 3D task with an easily understood target shape that is easily verified as correct.

Given the 3 by 3 grid, the box creation method is to cut off the corner squares and fold up the four flaps to be perpendicular to the central square. This must be done using the following steps:

1. Select two opposite corners of the 3 by 3 array and delete them. Opposite corners must be selected because selecting and deleting corners on the same row (eg. bottom left and bottom right) will cause the deletion all of the vertices of the bottom row.

2. Select the other pair of opposite corners of the 3 by 3 array and delete them, giving the plus-sign shape shown in figure 5.4.

3. Select the rightmost column of vertices and move them so that the rightmost quad stands perpendicular to the plane of the remaining squares, as if folding up a flap of a piece of paper, as in figure 5.4. The origin of the coordinate system is at the top right inner corner of the plus sign, and is marked by a 3D axis. Folding this flap first helps ensure perpendicularity, because it is located at the origin, and the Z axis line can be used to align the edge of the quad at its destination. The height of this flap must be 1 meter, which can be measured using grids, surface rulers, or contours.

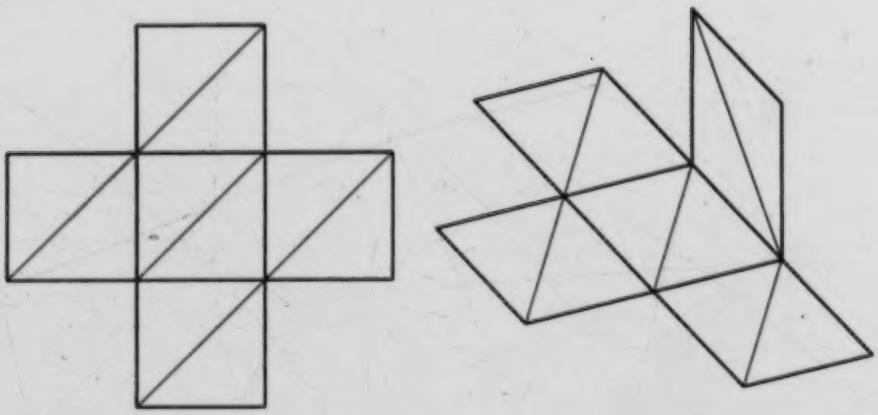


Figure 5.4:
Left: Task 2 **after** deleting the four corner squares.
Right: The **right** square has been folded up perpendicular to the central square.

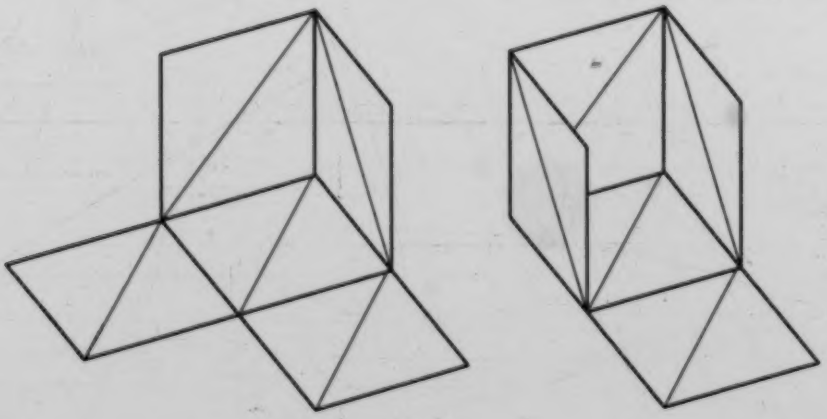


Figure 5.5:
Left: Task 2 **after** folding up the top square perpendicular to the central square to meet the right square.
Right: The **left** square has been folded up to meet the to square.

4. Select nothing, then select the topmost row of vertices and fold this top flap so that the right corner meets the left corner of the previous flap, as shown in figure 5.5. These flaps meet at the Z axis line. Height can be verified using the contour lines, surface rulers, and by verifying that the corners meet.

5. Select nothing, then select the leftmost row of vertices and fold this left flap so that the top corner meets the left corner of the previous flap, as shown in figure 5.5. Verify the height by contours, rulers, or matching corners.

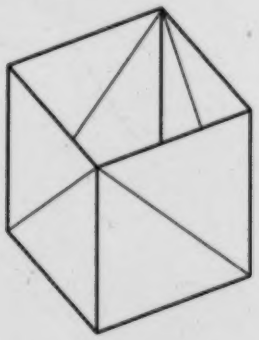


Figure 5.6: The completed task 2.

6. Select nothing, then select the bottommost row of vertices and fold this bottom flap so that the two corners meet the corners of the previous and first flaps, as shown in figure 5.6. Verify the height by contours, rulers, or matching corners.

Each flap is folded in two steps in THRED and the One-Handed interface. First, the vertices are moved up 1 meter by performing a line-constrained reshape along the Z axis, and stopping at the 1 meter contour line. Second, the vertices are moved to the perpendicular position by performing a second line-constrained reshape along the X or Y axis, as appropriate.

In the mouse-based interface, the task is a little simpler. In the *Top* orthographic view, the surface appears as in the left image of figure 5.4. The top vertices are selected in this window, and then they are moved up 1 meter in the *Front* window and left 1 meter in the *Right* window.

For the left and right flaps, the vertices are selected in the *Top* window, and then moved directly to the target position in the *Front* window using a 2D operation, because they are viewed edge-on in this window. The previously-folded top flap can be used as an alignment target. A similar operation can be used for the bottom flap in the *Right* window.

5.4 Experimental Structure

The initial structure of the experiment was to have a number of experimental subjects create a simple free-form surface using each of the three interfaces. This is a complete within-subjects design [59, 44], in which each subject gets all the treatments.

With this idea in mind, a pilot study using lab members was conducted, and it became clear that a significant amount of time would have to be spent in training the subject on each interface. Because of the large amount of learning involved, there was a worry that subjects would become overwhelmed with the task, and would be unable to feel enough confidence in themselves to make significant progress.

A second issue that concerned me was *asymmetric transfer*, which is a psychological effect of learning in which training and practice on one system adversely affects a subject's performance on a subsequent system. The way that this works is that a strategy learned in one condition of a within-subjects design experiment is inappropriately carried over and used in another condition of the experiment. For example, this might occur in a problem-solving experiment where two strategies A and B are compared. If A is used first, subjects may be unwilling to give up strategy A in order to use strategy B, since A is perceived to be a winner.

E. C. Poulton [50, 51] in particular warns against the use of within-subjects designs. While he acknowledges that balancing techniques control practice effects, he argues that they do not eliminate the asymmetric transfer problem. He maintains that the best way to deal with this problem is to conduct a second between-subjects design to document any asymmetric transfer occurring in the within-subjects design. Another way of dealing with this is to ensure that the presentation order of the treatments is randomized so that for the first block of trials, comparisons can be made between subjects. In effect, the idea is to compare the trials as if they were a between-subjects design [59].

One way of dealing with the strategy issue is to make sure that the subjects do not have to discover the strategy for themselves. The training regime can introduce both the method for using the tool and the method for best solving the problem using the tool. This also gets around the problem of subjects "cheating" and using strategies that worked well on the previous interface, because it does not require the use of non-optimal strategies.

To deal with the issue of too much learning, interfaces were compared in three pairs; Mouse-based and One-Handed, Mouse-based and Two-Handed, and One-Handed and Two-Handed. This lowers the amount of complexity that each subject has to deal with, and also allows more time for the subject to become proficient at doing the task.

The experiment was conceived as a counterbalanced incomplete within-subjects design. The term incomplete refers to the fact that no subject was exposed to the complete set of conditions. There were three groups of conditions, each group consisting of one of the pairs of interfaces outlined above. Within each group, subjects were randomly assigned to one of the two possible presentation orders, and these assignments were balanced, so that an interface preceded the other for half the subjects, and followed it for the other half.

5.5 Subjects

There were a total of 18 subjects in this experiment. Experimental volunteers were solicited from two second year mechanical and civil engineering drafting/computer-aided design courses. At the time of the experiment, all students had completed the term's course work in these classes. Volunteers were given $20 for their participation in the experiment.

Volunteers were first asked to read a tutorial. This tutorial, which appears in Appendix B, was either handed out in the class, or was picked up by the subject. The tutorial introduces the two interfaces they are to use, and explains the geometry, display, and interface commands. The paper makes it clear that there is a common geometry system, and that there are two interfaces. The second interface is presented more briefly than the first, assuming that the reader has gotten comfortable with the concepts, and only needs the details of the other interface.

Each tutorial was 8 or 9 pages long, and it seemed that few subjects read it, as few made any comments of recognition when the interfaces were demonstrated.

However, the tutorial papers acted as a random assignment mechanism, because there are three types of tutorials introducing a unique pair of interfaces. Each copy was sequentially numbered, and then all the copies were randomly ordered. As the subject received a tutorial paper, their contact information and tutorial number was recorded. No selection or assignment to experimental conditions took place based on the subject's characteristics.

5.6 Materials and Equipment

This experiment took place in General Services Building 642, using a Silicon Graphics 150MHz Crimson RealityEngine with a 21 inch screen. The trackers were a pair of Polhemus Isotraks synchronized at an update frequency of 30Hz, and driven by other computers on the local ethernet. The total tracker lag was about 100 milliseconds. The subject sat in a comfortable, padded chair with padded armrests, and with the armrest height approximately 1 inch below the height of the tabletop. The keyboard, mouse, and CRT were all mounted directly on this table without any height adjustments except for the mousepad. The subject would sit directly in front of the console for all practices and trials. Subjects were encouraged to make themselves as comfortable as possible, and to adjust their workspace as much as they deemed necessary.

All room lights were on for the experiment, delivering low to moderate lighting, as approximately half of the fixtures had working lights in them. The brightness of the room was approximately the brightness of a clear day just before sunset.

5.7 Experimental Steps

This section describes what a subject is to do during the entire course of the experiment. It includes answering questions on a questionnaire, receiving training, practicing on the task, and performing task trials.

5.7.1 Survey Part 1

When the subject arrived in the experiment room, they were asked to fill in the first four pages of a questionnaire, as shown in Appendix C.

Survey Page 1

Page 1 asks for student ID, birth date, sex, and program of study, and asks the subject to fill in a series of questions about hand preference.

The hand preference questionnaire is known as the Edinburgh Inventory, developed and characterized by Oldfield [48]. It asks the subject to indicate the preference in the use of hands in 10 manual activities such as writing, drawing, throwing, and so on. The tasks are listed in a table with two columns of check boxes labeled *left* and *right*, and the survey scheme is for the subject to place a mark in the appropriate box for the hand preferred in each task. Where the the prefetence is so strong that they would never try to use the other hand unless absolutely forced to, they are asked to put two marks in the check box, and if they are really indifferent, to put a mark in each box. Oldfield's paper presents a method of scoring this survey to generate a *Laterality Quotient* (LQ), which ranges from -100 for strong left-handedness to 100 for strong right-handedness. Based on a survey of thousands of Edinburgh University students, a ranking of LQ values in the general population can be used to generate a decile rank that is uniformly distributed. There are separate decile rank scales for left-handers and for right-handers, to account for the 10-to-1 more frequent occurrence of right-handers.

Survey Page 2

Page 2 asks the subject to report the amount of experience they have in various skills relevant to using computers and designing geometric objects. These questions ask subjects to account for regular use on a weekly or more frequent basis, as opposed to casual use, so as to evaluate their current level of practice. The skills asked about were the following:

1. General computer use.
2. Mouse use.
3. AutoCAD use.
4. Other CAD tool use.
5. Freehand drawing and drafting.
6. 6 DOF tracking system use.
7. 3D graphics programming and design.

Survey Pages 3 and 4

Pages 3 and 4 ask the subject to establish a baseline rating for the amount of pain and fatigue they are feeling prior to starting the training and trials. The rating is collected on a visual scale 10 centimeters wide, with a tick mark at each centimeter numbered from 0 to 10 as shown in figure 5.7. A rating of 0 indicates no pain or fatigue, and a 10 indicates worst possible pain or fatigue. The subject reports a rating by marking a line through the horizontal line of the scale or by circling a tickmark.

The 10 following body areas are rated, as follows:

1. Eyes and face.
2. Neck.
3. Shoulders.
4. Back.
5. Left hand.
6. Left wrist.
7. Left arm.
8. Right hand.
9. Right wrist.
10. Right arm.

The portion of the body below the belt was not rated because it did not seem relevant.

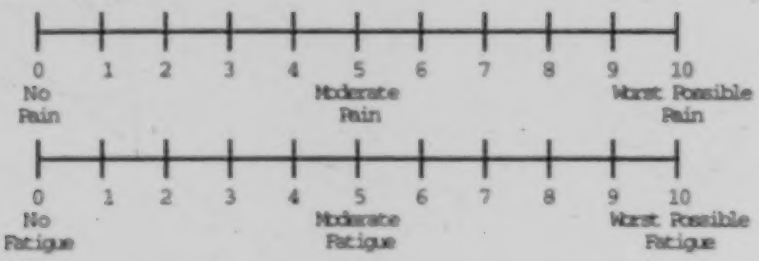

0 1 2 3 4 5 6 7 8 9 10
No Pain
Moderate Pain
Worst Possible Pain
0 1 2 3 4 5 6 7 8 9 10
No Fatigue
Moderate Fatigue
Worst Possible Fatigue

Figure 5.7: A pair of rating scales for pain and fatigue. The subject marks each by putting a line through the horizontal line of the scale or by circling a tickmark.

5.7.2 Demonstration of Interface A

After the subject completed the first four pages of the questionnaire, I demonstrated the first of the two interfaces that they were to use. As a demonstrational vehicle, I went through the steps required to perform task 1. I explained the following elementary tasks:

- How vertices are picked, the syntax of the currently selected set of vertices, and how to select nothing.
- How commands are executed from the menu.
- How to subdivide the array of squares in X and Y.
- How rulers are added to the surface.
- How vertices are reshaped. Subjects needed to be explicitly told that vertices were the objects being picked and edited, not polygons.
- How constrained reshaping is performed.
- How to travel about the scene.

Only operations used in the task were introduced and demonstrated. Several interface-specific instructions were needed for THRED and the one-handed interface:

The command to origin the cursor(s), and the command to align the spotlight were introduced. Because these were not needed for the task, I helped the subject with these.

Some subjects initially had difficulty with the sundial menu, because they would twist their menu hand into awkward positions. This happened because if they selected a menu item, they would leave their arm in that position, and then select another menu item. Their arm would move further and further away from a relaxed orientation as more menu items were selected. They were therefore advised to adopt one of the following strategies:

1. If you know where the menu is, *preload* the hand position required to select the menu by first moving in the opposite direction by an equivalent amount, and then start menu selection.

2. Once a menu item has been selected, always move your hand to a relaxed position.

Both strategies require that the subject make a compensatory movement to leave the hand in a relaxed position when the menu interaction is complete. The first strategy compensates before menu selection, and the second compensates after.

Some subjects also found that menu items on the lower left were difficult to hit, and they adopted a strategy of moving the bat down in order to move the shadow stick down into the desired menu item. This strategy didn't work, because the parallax induced by moving the menu down was cancelled by the hand's rotation of the Bat in the opposite direction about the elbow. Subjects were advised of this effect, and instructed to concentrate on rotating the bat with their fingertips, or to lift their elbow off the arm of the chair and thereby make it easier to get the desired orientation.

Some subjects were also confused about the snap arrow of the right cursor, so they were reminded that the right bat controlled the probe, not the snap arrow.

For the Alias clone, the window navigation commands needed to be introduced and explained, especially the disappearing cursor feature of the title bar commands.

5.7.3 Practice Task 1 with Interface A

In the next phase of the experiment, the subjects were asked to take a seat at the console, to get comfortable, and to practice the first task once to produce a correct result. This practice round was not timed. While the subjects were getting comfortable, I would make sure that the cursor(s) were correctly centered and the spotlight was aligned along the line of sight for THRED and the One-Handed interface.

In practicing the task, I would tell the subject step by step what to do next, and would coach them on the relevant details of using the interface. If they made a mistake, I told them how to correct the error. In the rare event of a total disaster, the subject would restart the practice round. All but one subject was able to complete the task successfully in one practice round.

5.7.4 Perform Task 1 with Interface A

With the practice successfully completed, the subjects were asked to complete 5 timed trials of the task with the following proviso in mind:

Go as fast as possible but make no errors.

I elaborated on this by informing the subject that little blunders along the way were perfectly OK, but that the requirement was to produce a correct model at the end. Therefore, any errors induced in the model must be corrected before the trial is over.

The protocol for timing was that I would signal the subject that I was ready to start the clock. When they were ready to start, they signaled me, and would immediately start the task as I started the clock. When they had completed the task and believed that the result was correct, they would signal me and I would stop the clock, record the time elapsed and tell them the time elapsed. I would also inspect for errors. If there was an error I would tell the subject about it, and explain what had gone wrong to make that error occur. Errors were recorded for any trial in which they occurred.

Between each trial was at least a 30 second rest before the next trial started. I would also coach the subject on using the interface and performing the task at this time.

If an utter disaster occurred during the course of a trial such as a bug in the software, or a mistake that made the surface difficult to reshape back into a useful condition, the trial was discarded and started again. In the case of a subject error, an error was recorded for the subsequent trial.

5.7.5 Task 1 Using Interface B

I next would demonstrate the steps needed to complete task 1 using interface B, concentrating on introducing the relevant features of the new interface. This demonstration took less time because the task was already familiar.

Next, the subject would practice under the same conditions outlined in section 5.7.3, and would perform 5 trials using the protocol of section 5.7.4.

5.7.6 Survey Pages 5 Through 8

Following the 10 trials of task 1, the second part of the survey was conducted, comprising 4 more pages of ratings. The first page (page 5) inquired about the subjects feelings of familiarity and expertise with the tools and the task. Pages 5 and 6 then continued with preference ratings for the two tools, both as a whole, and by analyzing their components. The next two pages (7 and 8) repeated the pain and fatigue survey from pages 3 and 4 (section 5.7.1).

The 4 familiarity and expertise questions queried the subject's self-rating in the following areas:

1. AutoCAD.
2. Interface A.
3. Interface B.
4. Creating Regular 3 by 3 grid.

AutoCAD was used to get an indication of how much more or less expert the subject felt about a standard tool in which they all had 3 months of formal training. Instead of "Interface A" and "Interface B", each subject's survey was customized for the presentation order of this subject's trials using the actual names of the interfaces, as follows:

- Mouse-Based Interface
- One-Handed 3D Interface
- Two-Handed 3D Interface

Each of these questions used a single scale identical in size to the pain scale shown in figure 5.7. For the familiarity and expertise questions, the values 0, 5, and 10 were labeled as follows:

0 Very Low Familiarity and Expertise

5 Moderate Familiarity and Expertise

10 Very High Familiarity and Expertise

The survey then asked for 14 preference ratings for the two tools, both as a whole, and by analyzing their components. The first two questions asked for a preference rating for Interface A and Interface B (customized as above). This rating used a 10 point scale with the values 0, 5, and 10 labeled as follows:

0 Strongly Dislike

5 Neither Like Nor Dislike

10 Strongly Like

Questions 3 through 14 asked for interface-to-interface comparisons using the scale shown in figure 5.8. This scale ranges from -5 to 5, with 0 indicating no preference. The left end of the scale is labeled with Interface A, and the right end is labeled with Interface B. Again, the scales are customized for the actual interfaces in the actual presentation order (One-Handed followed by Mouse in figure 5.8).

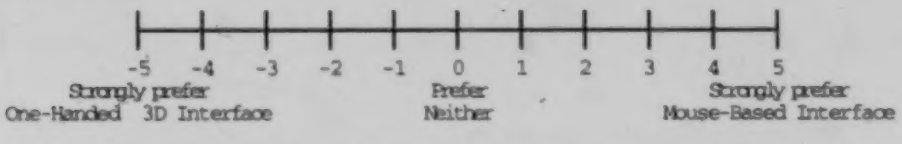


Figure 5.8: A rating scale for comparison between two interfaces.

Question 3 asks for a bulk comparison of Interface A versus Interface B. Questions 4 and 5 ask for a bulk comparisons of AutoCAD versus Interface A and Interface B, respectively (AutoCAD on the negative side for both questions).

Questions 6 through 14 ask for comparative preferences for the following 9 editor features:

6. Vertex Selection.
7. Moving and reshaping vertices without constraints.
8. Moving and reshaping vertices with constraints.
9. Activating Rulers.
10. View Control.
11. Examining Surface.
12. Examining Rulers.
13. Menu Appearance.
14. Menu Selection.

The following table summarizes the presentation of pages 5 and 6.

Number	Question type	Subject	Scale
1	Familiarity	AutoCAD	0 .. 10
2	Familiarity	Interface A	0 .. 10
3	Familiarity	Interface B	0 .. 10
4	Familiarity	Task 1	0 .. 10
1	Preference	Interface A	0 .. 10
2	Preference	Interface B	0 .. 10
3	Preference	Interface A vs. Interface B	-5 .. 5
4	Preference	AutoCAD vs. Interface A	-5 .. 5
5	Preference	AutoCAD vs. Interface B	-5 .. 5
6..14	Detailed Preference	Interface A vs. Interface B	-5 .. 5

5.7.7 Task 2 Using Interface B

After the subject completed questionnaire pages 5 through 8 and taken a refreshment break, subjects started on task 2 using interface B. This reversed the presentation order of interfaces from the trials for task 1, balancing for practice effects, and eliminating any bias that may result from a fixed presentation order.

In training on task 2 using interface B, the only additional command for task 2 is deletion of quads. Constrained reshaping is a little more involved than in task 1, because it occurs along more than one axis during the task. Also, reorientation of the surface to access its different parts is a little more involved, especially during the final inspection.

The subject then practiced performing the task once, during which I would coach them on the next step to take. By this time, the subject was familiar enough with the interface that only task coaching was necessary.

Next, subjects performed a trial of task 1 immediately followed by task 2. The timing protocol for this was that when the subject finished task 1, they would signal me, and continue onto task 2 without waiting for an acknowledgement from me. Subjects signaled the end of the trial as usual.

The original plan for the second set of trials was to perform task 1 followed by task 2 in all trials for both interfaces. However, because each of these trials took between 3 and 5 minutes, it became clear that not enough time was available to complete all the trials.

The protocol was changed so that a trial would consist of task 1 followed by task 2 until the time to complete task 1 was within 10 seconds of the best completion time in the previous block of trials. Subsequent trials would consist of just task 2 starting off from the window configuration that existed at the completion of task 1. Under this protocol, most subjects performed 2 trials of task 1 under each interface in the second block.

This is not an ideal protocol, since a different number of task 1 trials are used per subject in the second block of experiments. However it solved the practical difficulties of scheduling subjects, because I told subjects that the entire experiment would take 2.5 hours. I wanted to avoid two unpleasant situations – subjects feeling upset that the experiment was running too long, and having

one subject's experiment spill over into the time slot of the next subject. On the other hand, I wanted to verify that the subject's best time with an interface was not a fluke, so trying task 1 until the best time was approximated offered assurance that chance was not at play. Using a fixed number of task 1 trials did not offer this assurance.

In summary, the events for interface B were one complete demonstration of task 2 by me, followed by one practice session of task 2 by the subject, followed by 5 timed trials. The timed trials consisted of task 1 followed by task 2 until the task 1 time was approximately the same as the best task 1 time on interface B from the first block of trials.

5.7.8 Task 2 Using Interface A

The last set of trials used interface A, consisting of one complete demonstration of task 2 by me, followed by 5 timed trials by the subject. The timed trials consisted of task 1 followed by task 2 until the task 1 time was approximately the same as the best task 1 time on interface A from the first block of trials. The practice round was not used because the subject was by this time thoroughly familiar with the task and the interface.

5.7.9 Survey Pages 9 Through 12

The final 4 pages of the questionnaire asked the same questions as pages 5 through 8, with two important exceptions. First, the questionnaire asked 3 familiarity and expertise questions:

1. Interface B.
2. Interface A.
3. Folding a box.

The AutoCAD question does not appear a second time, and the last expertise question asks for a self-rating of task 2 expertise.

Second, the presentation order of the interfaces in pages 9 and 10 of the questionnaire is reversed from pages 5 and 6. Also, order is reversed in all of the preference questions, a fact which one subject failed to notice. The following table summarizes the presentation of pages 9 and 10.

Number	Question type	Subject	Scale
1	Familiarity	Interface B	0 .. 10
2	Familiarity	Interface A	0 .. 10
3	Familiarity	Task 2	0 .. 10
1	Preference	Interface B	0 .. 10
2	Preference	Interface A	0 .. 10
3	Preference	Interface B vs. Interface A	-5 .. 5
4	Preference	AutoCAD vs. Interface B	-5 .. 5
5	Preference	AutoCAD vs. Interface A	-5 .. 5
6..14	Detailed Preference	Interface B vs. Interface A	-5 .. 5

Pages 11 and 12 presented the third and final pain and fatigue questionnaire.

Finally, after completing the questionnaire, I asked the subject for any free-form comments they may have had, and I noted these briefly on the back of the questionnaire.

5.8 Summary

In summary, an experimental subject followed these steps to perform one complete experiment.

1. Fill in page 1 of the survey, giving demographic and handedness data.
2. Fill in page 2, describing experience in computers, 2D and 6 DOF spatial input devices, CAD systems, drawing, and drafting.
3. Fill in pages 3 and 4, rating pain and fatigue for 10 parts of the upper body.
4. Receive training for task 1 on Interface A.
5. Practice one iteration of task 1 using Interface A.
6. Perform 5 timed trials of task 1 using Interface A.
7. Receive training for task 1 on Interface B.
8. Practice one iteration of task 1 using Interface B.
9. Perform 5 timed trials of task 1 using Interface B.
10. Fill in pages 5 and 6, rating familiarity (4 questions) and preferences (14 questions).
11. Fill in pages 7 and 8, rating pain and fatigue.
12. Rest break.
13. Receive training for task 2 on Interface B.

14. Practice one iteration of task 2 using Interface B.

15. Perform 5 timed trials of task 1 + task 2 using Interface B.

16. Receive training for task 2 on Interface A.

17. Perform 5 timed trials of task 1 + task 2 using Interface A.

18. Fill in pages 9 and 10, rating familiarity (3 questions) and preferences (14 questions).

19. Fill in pages 11 and 12, rating pain and fatigue.

20. Make free-form comments.

Subjects were assigned randomly to one of the 6 following conditions, 3 subjects per condition, for a total of 18 subjects.

Interface A	Interface B
Mouse	One-Handed
Mouse	Two-Handed
One-Handed	Mouse
One-Handed	Two-Handed
Two-Handed	Mouse
Two-Handed	One-Handed

Chapter 6

Experimental Results – Demographics and Training

This chapter explains the results of the experiment described in Chapter5. The demographic information will be described first, followed by the times for each segment of the experiment. Chapter 7 covers the results of the pain and fatigue survey. Chapter 8 covers the preferences survey, and finally the timing results are covered in chapter 9.

6.1 Demographics

There were 16 male and 2 female subjects, ranging in age from 19 to 28 years. All but 3 subjects were between the ages of 19 and 21 inclusive. Dividing by program of study, 12 were enrolled in Mechanical Engineering, and 4 were in Business, and 2 were in Industrial Design. Each condition representing a pair of interfaces (6 subjects) had 4 mechanical engineers and 1 business student (this was a random occurrence).

Applying Oldfield's [48] calculations to determine Laterality Quotient (LQ), and looking up the LQs in Oldfield's decile rank table determined the handedness decile for each subject. Surprisingly, almost 40% of the subjects turned out to be strongly right-handed, ranking in the 10th decile. The table 6.1 lists the populated deciles and the population of each, with the negative decile indicating a left-handed subject. Oldfield's decile ranking of left-handers accounts for the fact that left-handers commonly adapt their manual skills to adapt to a right-handed world, so a -1 decile indicates ambidexterity.

Decile	Population	Decile	Population
10	7	4	4
7	2	3	1
6	1	2	1
5	1	-1	1

Table 6.1: List of handedness decile ranks and their populations.

Table 6.2 shows the assignment of subjects to condition, not accounting for order of interfaces.

Condition	MechEng	Business	Design	Sex	Mean Hand Decile
MouseOne	4	2		1F, 5M	5.7
MouseTwo	4	1	1	1F, 5M	7.5
OneTwo	4	1	1	6M	6.0

Table 6.2: Demographic breakdown of subjects by condition.

Despite the heavy skewness in the demographics, there is a fairly uniform assignment of subject characteristics to conditions. No condition has significantly more strong right-handers, females, or students of a particular program.

6.1.1 Experience Questions

Because of the source of subjects, everyone had at least three months of computer experience. In fact, only 4 subjects had less than a year's computer experience. At the other end of the scale, 5 of the subjects had been using computers for almost half of their lives!

The mean amount of computer experience was 3.6 years, and the mean amount of mouse experience was 1.9 years. In all but 5 cases, the amount of mouse experience was equal to the amount of computer experience. The distribution of these measurements was approximately uniform.

AutoCAD experience was highly centralized, however, with all but one subject having 3 months of experience. Two other subjects had 3 and 12 months of experience in CADKey and 3D Studio, respectively, with the rest of the subjects unexposed to 3D CAD tools of any kind.

Subjects had 6.8 months of recent free-hand drawing, 2.8 months of recent drafting experience, and 2 weeks of programming or designing 3D objects. No subject had any prior exposure to 6 DOF input devices such as the Polhemus Isotrak, the Logitech Ultrasonic mouse, or Spaceball.

The only correlation of one experience rating to another is between mouse experience and computer experience. The other measures are not correlated in a statistically significant way.

Because the arithmetic mean is susceptible to the influence of extreme data, the means reported in this section are maximum-likelihood estimators that give less weight to data more than a standard deviation away from the mean. Tukey's Biweight estimator is used, which is supplied by SPSS [47]. In table 6.3, columns labeled *M-Est* use this estimation procedure instead of the arithmetic mean to report central tendency.

Experience Area	M-Est	Std Dev	Min	Max
Computer (Years)	3.6	3.3	0.3	9.6
Mouse (Years)	1.9	2.3	0.3	8.3
AutoCAD (Months)	3.0	0.7	3.0	6.0
Freehand Drawing (Months)	6.8	20.9	0	60.0
Drafting (Months)	2.8	7.2	0	27.0
3D Design (Months)	0.6	3.0	0	12.0

Table 6.3: Summary statistics outlining the subjects' experience.

6.2 Times

In recording the progress of the trials of this experiment, I noticed that a significant amount of time was being spent on non-trial activity. The initial assumption was that training, practice, and answering survey questions would take only a small proportion of the time allotted for the whole experiment. However, this was not the case and so after the sixth trial, I started collecting more careful records of the time spent throughout the course of the experiment. This consisted of recording the current time at the end of each activity. The four major classes of activity were *survey*, *training*, *practice*, and *trial*. Survey includes answering 4 pages of survey questions, and any extra rest time that the subject may have taken. There were three survey periods: *Start*, *Middle*, and *End*. There were four training sessions, one for each interface paired with each task for a subject. All but the last training period was followed by a practice period, and finally there were four trial periods. Table 6.4 displays the means and standard deviations of the 14 time periods of the experiment, in order of occurrence.

Period	M-Est	Std Dev	Min	Max
Survey pages 1..4	10:11	5:24	4:00	17:00
Training Interface A Task 1	15:21	4:16	9:00	24:00
Practice Interface A Task 1	11:30	3:07	6:00	16:30
Trials Interface A Task 1	18:41	5:34	12:30	27:12
Training Interface B Task 1	7:13	3:39	6:30	15:42
Practice Interface B Task 1	7:10	4:42	4:00	19:00
Trials Interface B Task 1	14:13	4:15	8:00	21:30
Survey pages 5..8	6:52	3:45	4:30	18:00
Training Interface B Task 2	6:19	1:08	4:00	7:36
Practice Interface B Task 2	6:10	2:18	3:00	10:00
Trials Interface B Task 2	22:07	7:11	10:54	32:00
Training Interface A Task 2	4:16	1:55	1:42	7:42
Trials Interface A Task 2	21:28	8:09	10:00	37:00
Survey pages 9..12	3:06	0:08	3:00	3:12
Total Experiment	2:02:19	21:05	2:14:00	3:10:00

Table 6.4: Stage-by-stage breakdown of experimental time spent.

As table 6.5 shows, about half of the time is spent on conducting trials, one quarter on training, and one sixth each on survey and practice. The numbers for answering the survey are not terribly interesting, because of the unstructured way in which that time is spent. Also, the mean time for the last survey derives from only two readings, because it was not diligently recorded. Similarly, the trial times are only indicative of the expected time for 5 trials, due to variables resulting from experimenter error and subject errors.

However, the training and practice times are of significant interest, because it gives a strong indication of the ease of use of the systems.

6.2.1 First Session Training Times

Hypothesis 1: First session training times for the One-Handed and Two-Handed systems exceed

Period Type	Sum of M-Estimates	Percent of Total
Survey	21:16	13
Training	36:19	22
Practice	26:01	16
Trial	1:18:05	49

Table 6.5: Mean times spent in experiment.

Mouse training times by less than a factor of 2.

The first session training time is the time spent to explain interface A on task 1. Table 6.6 shows the mean training times for the three interfaces. The one-handed case seems to give short training times close to the mouse case. Lumping the non-mouse cases yields the 4th line in the table.

Interface	M-Est	Std Dev	Number
Mouse	13:11	3:34	4
One	15:07	0:45	3
Two	20:26	4:02	3
One+Two	15:23	3:50	6

Table 6.6: First session training times reported by interface.

In fact, it seems that the non-mouse training times are only 17% longer than the mouse training times. An examination of the normality of the distributions of these two collections of times using a Shapiro-Wilks test yields a significance level for 6 degrees of freedom of .0855, indicating that these numbers are perhaps not normally distributed.

This means that the F test (analysis of variance) and the Student's T-Test are not valid due to the normality assumption of these tests. The alternative to the T-Test is the nonparametric Mann-Whitney U Test, which tests for differences between means by ranking all of the data, and using the distribution of the ranks into the two classes to determine a statistic. This test has a relative power of 95.5% compared to the T-Test, which means it will discriminate 95.5% of the means that the T-Test would find to be different [63].

Comparing the means with the hypothesis that the non-mouse case has a greater mean than the mouse case, the Mann-Whitney U test has a 1-tailed significance probability of 0.0816 after accounting for tied ranks, indicating that the differences in means could be the result of chance 8.16% of the time.

Comparing the mouse times to just the two-handed times, the training time is 55% larger, with a Mann-Whitney 1-tailed significance probability of 0.038.

One explanation for the difference could be that I intentionally took a longer time to train in the two-handed case, but the M-estimates would not be statistically close to each other if this were the case. Also, the two-handed and mouse variances are similar, which also indicates true variability in training time. In fact, the two-handed case had an outlyer subject who took 24 minutes to train, which is still less than a factor of two times the mouse average training time.

The major reason why training the non-mouse interfaces take longer is that they are totally new

to the subject. Training on the mouse-based interface takes less time because one can rely on a subject's prior experience.

In any case, since there is only bare evidence that the M-estimates are different at all, I can confidently state that training on the one-handed and two-handed systems takes only 50% longer.

6.2.2 First Session Practice Times

Hypothesis 2: First session practice times for all systems are equal.

This is the time allotted for the subject to successfully complete one practice trial with coaching for Interface A on Task 1. Again, table 6.7 compares mouse to non-mouse interfaces.

Interface	M-Est	Std Dev	Number	S-W Sig
Mouse	9:45	3:16	4	
One	12:21	4:15	3	
Two	11:45	1:19	3	
One+Two	12:34	2:49	6	.9394

Table 6.7: First session practice times reported by interface.

In this case, the Shapiro-Wilks normality test does not indicate a departure from normality, so a T-Test can be used. Testing for the equality of means, the T-Test yields a 2-tailed significance of 0.225, indicating that these means are probably equal. A Mann-Whitney U test yields a similar result, with a 2-tailed probability of .2395. Statistically, then, it seems that there is no reason to reject the null hypothesis.

6.2.3 Second Session Training Times

Hypothesis 3: Subject's familiarity with Interface B strongly influences training time.

After the first trial block with Interface A on task 1, the subjects are trained on Interface B. Hypothesis 3 says that the more familiar the subject is with the interface, the less training time is needed. Counting presentation order, table 6.8 shows the six experimental conditions.

Interface A	B	B M-Est	B Familiar?	Comment
One	Two	6:30	Yes	Interface B is similar to A.
Two	One	7:00	Yes	Interface B is similar to A.
One	Mouse	6:42	Yes	
Two	Mouse	7:22	Yes	
Mouse	Two	13:44	No	
Mouse	One	15:42	No	

Table 6.8: Interface B training times reported by interface accounting for interface presentation order.

With the mouse-based interface, the trainer can rely on the subject's 1.9 years of mouse experience. In assuming familiarity, the trainer can simply explain the details of the new interface, because subject already knows how to operate the interaction techniques. Also, having just completed 5 trials, the subject is familiar with the task.

In the last 2 conditions above in table 6.8, Interface B is unfamiliar, so the interaction techniques must additionally be introduced and explained. Table 6.9 lumps the 4 familiar conditions in one group, and the 2 unfamiliar conditions in the other group.

Familiarity Condition	M-Est Time	Std Dev
Familiar	6:44	0:55
Unfamiliar	14:07	1:09

Table 6.9: Second session training times grouped into familiar and unfamiliar classes.

These data are again not normally distributed, so the Mann-Whitney U test yields a 1-tailed significance probability of 0.004, strongly indicating that the difference in means is statistically significant.

The small standard deviations for each familiarity condition indicate that the interface type is unimportant. Table 6.10 shows the means of the familiar interfaces. Checking the familiar interfaces for equality of means between interface type, a Mann-Whitney U test for a difference of means yields a significance probability of 0.434.

Condition	M-Est Time	Std Dev
Mouse	6:48	1:02
One+Two	6:45	0:21

Table 6.10: Second session training times for familiar interface B, broken down by interface type.

The likeness in training times for the familiar interfaces also indicates that there is no trainer bias in the training process.

Comparing the training times for Interface A with the training times for Interface B gives an indication of the distribution of training effort between the task, the surface modeling system, and the interface. Since the task is familiar for Interface B, and since the familiar times are the same regardless of interface, subtracting the time for familiar Interface B from familiar Interface A yields the added time needed to introduce the task and the surface modeling system:

$$FamiliarA - FamiliarB = Task\ and\ Surface\ Intro$$

$$13:11 - 6:44 = 6:27$$

Thus, about 6.5 minutes are devoted to task and surface modeling system training. This number should be taken with a grain of salt because of the large variances in the training time for familiar Interface A. A within-subjects comparison cannot be performed because a maximum of one interface is familiar per subject.

A similar analysis can be applied in order to estimate the amount of time needed to familiarize the subject with a brand-new interface style. Again, the task is no longer an issue because the subject has been doing it for 5 trials, although the task must still be trained for this interface.

$$UnFamiliarB - FamiliarB = New\ Style\ Instruction$$

$$14:07 - 6:44 = 7:23$$

Adding up these component times yields:

$$New\ Style\ Instruction + FamiliarB + Task\ and\ Surface\ Intro = UnFamiliarA$$

$$7:23 + 6:44 + 6:23 = 20:30$$

This is 5:07 minutes longer than the actual training time for the unfamiliar One-Handed and Two-Handed interfaces, indicating that there is a significant amount of parallel learning occurring in the training process.

With regard to hypothesis 1, the difference in training times between Mouse and non-Mouse is 2.08 times longer, or approximately a factor of two, as hypothesized.

6.2.4 Second Session Practice Times

Hypothesis 4: Subject's familiarity with Interface B strongly influences practice time.

Hypothesis 5: Second session practice times for the One-Handed and Two-Handed systems exceed Mouse practice times by less than a factor of 2.

Looking at the practice times by condition, we again have a distinct split of practice times based on the familiarity of the interface, as shown in table 6.11.

Interface A	B	B M-Est	B Familiar?
One	Two	7:00	Yes
Two	One	6:24	Yes
One	Mouse	4:00	Yes
Two	Mouse	6:09	Yes
Mouse	Two	14:42	No
Mouse	One	11:30	No

Table 6.11: Second session practice times reported by interface accounting for presentation order.

Table 6.12 displays the second session practice times divided into familiar and unfamiliar conditions.

Familiarity Condition B	M-Est Time	Std Dev
Familiar	5:00	1:21
Unfamiliar	13:28	4:00

Table 6.12: Second session practice times – familiar vs. unfamiliar.

The Shapiro-Wilks statistic indicates probably normal distribution (significance = 0.2281), so a T-Test can be applied. A Levene test for equality of variance gives a significance probability of 0.018, indicating unequal variances. The T-test with unequal variances yields a 1-tailed significance probability of 0.0125, strongly indicating that the difference in means is statistically significant.

Testing for the equality of the means for the Mouse condition to the means for the Familiar One-Handed or Two-Handed cases gives a 2-tailed probability of .181, indicating that the practice times for the two Familiar interface classes are the same. Together, these two analyses indicate that difference in practice times is due solely to the familiarity of the interface, confirming hypothesis 4.

Repeating the calculations from section 6.2.3 to estimate the time a subject takes to deal with just the task component of the practice session for task 1 yields:

$$FamiliarA - FamiliarB = Task\ 1\ Practice\ Time$$

$$9:45 - 5:00 = 4:45$$

Thus, almost 5 minutes of a subject's practice effort in the first practice session relate to learning the task. An estimate of the amount of time the subject spends in getting familiarized with the new interface is as follows:

$$UnFamiliarB - FamiliarB = New\ Style\ Practice$$

$$13:28 - 5:00 = 8:28$$

This is one minute longer than the training time for the new style. Adding up these component times yields:

$$New\ Style\ Practice + FamiliarB + Task\ 1\ Practice\ Time = UnFamiliarA$$

$$8:28 + 5:00 + 4:45 = 18:13$$

This is 4:45 minutes longer than the actual practice time for the unfamiliar One-Handed and Two-Handed interfaces, indicating parallelism in learning.

When the One-Handed or Two-Handed systems are unfamiliar, the practice time is (13 : 28/5 : 00) times as long, or a factor of 2.7 times as long, which is somewhat longer than expected based on the results for interface A. It seems that the narrower differences for Interface A practice arises out of the unfamiliarity of the task.

6.2.5 Third Session Training Times

Hypothesis 6: Third session training times for all interfaces are the same.

After completing the two sets of trials for creating a 3x3 grid and answering pages 5..8 of the survey, the subjects receive training on folding a box using Interface B. To save time, the training session starts with a regular 3x3 grid loaded from disk. By this time, the subject is about equally familiar with both interfaces. Because the strategy for completing Task 2 with the mouse-based interface is somewhat different than for the One-Handed or Two-Handed interfaces, there might be some concern about different means.

Table 6.13 shows the third session training times. We must apply a test for the hypothesis that the means are the same. The Shapiro-Wilks statistic indicates probably normal distribution (significance = 0.6131), so a T-Test can be applied. A Levene test for equality of variance gives a

Interface	M-Est	Std Dev	Number
Mouse	5:23	1:22	5
One	7:18	0:25	2
Two	6:24	0:20	4
One+Two	6:42	0:34	6

Table 6.13: Third session training times reported by interface.

significance probability of 0.105, indicating unequal variances. The T-Test with unequal variances yields a 2-tailed probability of 0.128 for the Mouse mean being different from the non-Mouse mean, indicating that they are probably not different.

Comparing the Mouse case to the One-Handed case, A Levene test for equality of variance gives a significance probability of 0.246, indicating equal variances. The T-Test with equal variances yields a 2-tailed probability of 0.145 for the Mouse mean being different from the One-Handed mean.

There is no statistical evidence that the means for training times for interface B on the box folding task are different.

It is interesting to compare to the previous training session. However, only the cases where the interfaces were familiar can be used, because the unfamiliar interfaces had longer session 2 training times. Compared to the familiar cases of the second session, the means are about the same, as shown in table 6.14.

Session	M-Est	Std Dev	Number
Familiar Session 2	6:44	0:55	7
Familiar Session 3	5:57	1:22	7
All Session 3	6:18	1:08	11

Table 6.14: Training session 2 compared to session 3.

To compare these practice times within subjects, a paired-samples T-Test could be used, but a Shapiro-Wilks statistic yields a significance level of 0.0384, indicating non-normality. The corresponding nonparametric test is the Wilcoxon matched-pairs signed-ranks test, which ranks matched pairs (the two practice times per subject in this case) and calculates a statistic based on the average rank for each case. This test yields a 2-tailed significance probability of .9165, indicating that the means are the same.

6.2.6 Third Session Practice Times

Hypothesis 7: Third session practice times for all interfaces are the same.

Interface	M-Est	Std Dev	Number
Mouse	7:03	2:00	5
One	8:22	2:18	2
Two	4:38	2:36	4

Table 6.15: Third session practice times reported by interface.

Table 6.15 shows the practice times split by condition. The means seem somewhat different, but the standard deviations are very large, and the distribution of these values is non-normal, as indicated by the Shapiro-Wilks significance for the Mouse case of 0.0749. Checking for equality of means between the Mouse and Two-Handed case, a Mann-Whitney U test for a difference of means yields a 2-tailed significance probability of 0.9021. Despite the apparently wide separation, the overlap of the distributions does not indicate that the means are different. Similarly, performing a Mann-Whitney U test to compare the One-Handed to the Two-Handed case yields a 2-tailed significance probability of 0.1649.

Compared to the second practice session (Interface B, Task 1), familiar cases must be chosen because the unfamiliar cases in session 2 take somewhat longer. Examining just the familiar cases, the means are shown in table 6.16.

Session	M-Est	Std Dev	Number
Familiar Session 2	6:01	1:21	7
Familiar Session 3	6:43	1:41	7
All Session 3	6:10	2:17	11

Table 6.16: Practice session 2 compared to session 3.

Both of these have Shapiro-Wilks statistics that indicate normality (.2281 and .1939), so a paired-samples T-Test can be used. This test yields a 2-tailed significance probability of 0.925, indicating that the means are the same.

6.2.7 Fourth Session Training Times

Hypothesis 8: Fourth session training times are influenced by the the similarity of the third session interface type.

Hypothesis 9: Fourth session training times are less than third session training times per subject.

Because the trainer can rely on strategies that have been previously introduced, One-Handed or Two-Handed interface training sessions following Two-Handed or One-Handed training sessions should be shorter than those involving the Mouse interface in either position. Unfortunately, only one training time was collected for a One,Two or Two,One condition. This training time was less than all of the other training times. The tests reported in table 6.17 exclude this single sample.

Interface	M-Est	Std Dev	Number
Mouse	4:57	2:18	4
One	5:14	1:31	3
Two	3:33	0:04	2
Two,One	1:42	0	1

Table 6.17: Fourth session training times reported by interface.

Checking for difference of means between the Mouse case and the Two-Handed case, using a Mann-Whitney U test yields a 2-tailed probability for different means of 0.5333. A Mann-Whitney

U test comparing the One-Handed against the Two-Handed case yields a 2-tailed probability of 0.200.

It seems that training on the Mouse-based interface either before or after has no effect on training times. There is no evidence either for or against hypothesis 8.

Session	M-Est	Std Dev	Number
All Session 3	6:18	1:07	11
All Session 4	4:16	1:55	10
Session 4 without Two,One	4:23	1:45	9

Table 6.18: Training session 3 compared to session 4.

Table 6.18 shows the training means for sessions 3 and 4. Both of these have Shapiro-Wilks statistics that indicate normality (.5892 and .6912), so a paired-samples T-Test can be used. This test yields a 1-tailed significance probability of 0.028, indicating that the means are indeed quite likely to be different.

To calculate the benefit of repeated exposure to the same task, the session 3 and session 4 training times can be compared. The latter training time is shorter because the trainer can assume that the subject knows how the task and the interface work, and just needs to know the details of doing it on the other interface. By this time both interfaces are about equally familiar, so the difference in times is the amount of time it takes to explain what the task is.

$$FamiliarB - FamiliarA = Task\ Introduction\ Time$$

$$6:18 - 4:23 = 1:55$$

6.2.8 Discussion of Training and Practice

Training times in this experiment are influenced by four parameters:

1. The task.
2. Previous exposure to the task.
3. Previous general familiarity with the interface style.
4. Previous experience with a similar interface.

The influence of the trainer no doubt exists, but evidence of bias toward one interface does not seem to exist. There is no difference in mean training times or standard deviations for like interfaces during the same training session. For example, training session 2 on any unfamiliar interface takes the same amount of time, on average. Similarly, the lack of a statistically significant difference between interface types in sessions 3 and 4 indicate a lack of trainer bias.

The most significant influence on training time is the subject's familiarity with the interface. Subjects do not have to be told in too much detail how the menus operate in the Mouse-based

interface, and they are familiar with the idea of the 3-view drawing. Familiarity is of two types – general familiarity with the interface style, and specific familiarity with a particular interface. Mixed in with this is familiarity with the underlying surface operations. It is difficult to separate training times into separate components because of this confounding.

Session	Class	Time	Class	Time	Difference
1	Mouse	13:11	Non-Mouse	15:23	2:12
2	Familiar	6:44	Unfamiliar	14:07	7:23
3	All	6:18			
4	All	4:23			1:55

Table 6.19: A summary of training times separated into appropriate classes for each session. The last line shows the difference between session 3 and session 4.

Examining table 6.19, the last 2 training sessions indicate that task introduction can be broken into task introduction and task training. Thus, session 2 consists only of task training, since task introduction has already taken place in session 1. Comparing the Non-Mouse session 1 time to the Unfamiliar session 2 time yields a difference of 1:16, which corresponds approximately to the 1:55 difference in training times between sessions 3 and 4. Comparing the Mouse session 1 time to the Familiar session 2 time yields a difference of 6:27. This difference could be broken into task introduction, and surface modeling introduction.

$$Task\ and\ Surface\ Intro = Task\ 1\ Intro + Surface\ Modeling\ Intro$$

$$6:27 = 1:16 + 5:11$$

The difference between session 1 Mouse and Non-Mouse times is 2:12, indicating a minimum premium for introducing a totally new interaction style. Comparing session 2 times, the Familiar interfaces took 7:12 less than the Unfamiliar ones, indicating that a maximum premium for introducing the new style is about 7 minutes.

Table 6.20 summarizes these components.

Time Element	Time
Task 1 Introduction Time	1:16
Task 2 Introduction Time	1:55
Task Training Time	4:23
Surface Modeling Introduction Time	5:11
Non-Mouse Interface Introduction Premium	7:23

Table 6.20: An estimate of component times that contribute to the total training time.

For practice times, the same factors as for training come into play, as table 6.21 shows. There was no fourth practice session, so the practice on task 2 cannot be broken into subcomponents. Table 6.22 summarizes the components.

The interpretation of these results is that introducing a new interface style using a simple task takes about 16 minutes longer than for a mouse-based system, assuming that the learner knows how

Session	Class	Time	Class	Time	Difference
1	Mouse	9:45	Non-Mouse	12:34	2:49
2	Familiar	5:00	Unfamiliar	13:28	8:28
3	All	6:11			

Table 6.21: A summary of practice times separated into appropriate classes for each session.

Time Element	Time
Task Practice Time	5:00
Surface Modeling Learning Time	4:45
Non-Mouse Interface Learning Premium	8:28

Table 6.22: An estimate of component times that contribute to the total practice time.

the basic operations of the interface work. This is not ideal, of course, because one would prefer that the user be able to use this new style with no training, or just by reading the manual.

However, this is a short training and practice time. In my experience of training first-time Macintosh users in CMPUT 161, the first hour or so is spent in training everyone in a lab to boot, initialize two diskettes and back up two software diskettes to these initialized disks. Of course, the subject population is different in my experiment, but some of the same basic issues arise in both situations. For example, some Mac users must be told to re-center the mouse, as some THRED users must be told to re-center the orientation of their hand after menu selection.

The wider disparity of practice times as compared to training times indicates that there is a certain amount of manual skill that must be learned by dint of practice. For the mouse-based interface, 1.9 years of mouse experience is at the subject's disposal, while there is no 6 DOF tracker experience to draw on. As a result, subjects take 2.7 times as long on the unfamiliar interfaces as on the familiar ones.

Chapter 7

Experimental Results – Pain and Fatigue

This chapter explains the results of the experiment described in Chapter 5 that relate to the Pain and Fatigue survey. The companion to this chapter is Appendix A, which contains the detailed statistical treatment of the survey results. Identical section titles and numbers are used so that the correspondence is easy to find. For completeness, the tables that appear in this chapter are duplicated in Appendix A.

Despite the fact that pain and fatigue are wholly subjective entities, measuring them seems to be worthwhile because it is an important question whether THRED is good ergonomically. All subjects sat in the same padded chair with padded armrests, at the same workstation with the same trackers, so the only variable in the subject's physical environment was the experimental condition. Of course, the other big variable is the subject. Unfortunately, there is not a true *control* condition to compare the experimental conditions against, such as a set of trials where two different mouse-based interfaces were used. This would help control for subject variability, giving a set of pain and fatigue ratings for a known pair of interface styles.

A total of three pain and three fatigue ratings were collected per body part over the course of the experiment. The first rating can used as a zero point for the two following ratings, thus controlling for pre-existing conditions. For example, one subject has braces installed the day before the experiment, and consequently rated eye/face pain at 7 throughout the experiment.

A total of 1080 raw pain or fatigue ratings were collected during this experiment. Dividing by the number of subjects per condition, there are 180 raw pain or fatigue means:

$$10\ \textit{Body Parts} \times 3\ \textit{Conditions} \times \textit{Pain or Fatigue} \times 3\ \textit{Surveys}$$

However, examining these raw numbers is not terribly enlightening because of the pre-existing conditions implicit in the raw ratings. Instead, the *normalized scores* will be reported, which are calculated by subtracting the initial rating from any subsequent rating of the same item. This cuts the number of means to report to 120. These will be further broken down into reports for pain and for fatigue, divided by body area:

- Head and Torso
- Left Arm and Hand
- Right Arm and Hand

The use of normalized scores means that negative scores are possible, due to variability in second and third raw ratings after a nonzero initial rating.

Three general hypotheses will be tested with these data:

Hypothesis 10: Pain or fatigue scores increase by less than 1 point midway through the experiment.

Hypothesis 11: Pain or fatigue scores increase from the initial ratings by less than 1 at the end of the experiment.

Hypothesis 12: End Pain or fatigue scores are the same as the midway scores.

Hypothesis 13: Pain or fatigue scores are the same for all interface conditions.

For normally distributed data, hypotheses 10 and 11 can be tested using a T-Test and examining the 95% confidence interval. The null hypothesis states that the mean equals 0, and if this hypothesis is true at a 5% level, then the 95% confidence interval will include 0.

Hypothesis 12 can be tested by performing a paired-sample T-Test between the midway scores and the end scores if both scores are statistically different from 0.

Hypothesis 13 can be tested by performing pairwise T-Tests between the scores for each condition.

For non-normally distributed data, nonparametric tests must be used. Hypotheses 10 and 11 can be tested using a Wilcoxon matched-pairs signed-ranks test. For these hypotheses, the ratings will be compared pairwise against the value 1, while for the null hypothesis, the ratings will be compared pairwise against 0. The comparison of the scores against 0 is identical to comparing the first raw rating to the midway raw rating or end raw rating, as appropriate.

Hypothesis 12 can be tested by performing a Wilcoxon test between the midway scores and the end scores, and hypothesis 13 can be tested with the Mann-Whitney U test.

7.1 Head and Torso Fatigue

Table 7.1 shows the fatigue scores for head and torso at the midpoint and the end of the experiment for all of the subjects. The columns labeled S-W is the significance probability of a Shapiro-Wilks normality test. Small probabilities indicate a departure from normality.

Fatigue All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	-.0716	1.2154	.05	.0278	1.9847	.01
Neck	.1235	.7334	.01	.0661	.7883	.03
Shoulders	.1975	1.3216	.01	.3827	1.4271	.01
Back	.1111	1.0870	.01	.0802	.8428	.01

Table 7.1: All-subject fatigue statistics for head and torso.

None of these variables have large enough Shapiro-Wilks probabilities, so nonparametric tests must be used. For each body area, the tables in Appendix A list the results of statistical tests for the null hypothesis for midway and end scores. First, the null hypothesis for Midway and End scores is tested, using a 2-tailed probability, because the test is for whether the actual mean is different from 0 in either the positive or negative direction. Hypothesis 12 (that midway and end scores are the same) can uses the results of the previous statistical tests. A significance level of 0.05 is used.

Table A.1 shows that the probabilities for the Midway=0 hypothesis are all greater than 0.05, indicating that the Midway scores are not likely different from 0. The same result exists for the End scores, indicating that they are also not likely different from 0. The acceptance of the null hypothesis for midway and end scores indicates that these two sets of scores do not differ from each other.

Tables A.2 through A.6 indicate that for the MouseOne, MouseTwo and OneTwo conditions, the null hypothesis should be accepted for all of the means. That is, for each condition and each body area, the mean is not different from 0 by a statistically significant margin. Since none of tests give evidence against the null hypothesis, we can assume that hypotheses 10, 11, and 12 can be accepted.

In dealing with hypothesis 13, comparisons between means for each condition should be made. A total of three comparisons can be made, for each pair of conditions using a Mann-Whitney test or a T-Test if the data are normally distributed. The headings of table 7.2 indicate condition pairings:

- M1-M2 = MouseOne - MouseTwo
- M1-12 = MouseOne - OneTwo
- M2-12 = MouseTwo - OneTwo

Table 7.2 presents tests for difference of Midway fatigue scores, with means briefly summarized. The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Fatigue Midway	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Eyes/Face	.20	-.20	-.18	.511T	.932	.342
Neck	.03	0.00	.06	.669T	.708T	.515T
Shoulders	.42	-.04	.70	.032	.849	.032
Back	.29	-.25	.37	.340	.263	.045

Table 7.2: Midway fatigue comparisons between conditions for head and torso.

For the Midway fatigue scores, the Eyes/Face and Neck scores are similar for all three conditions. The Shoulder scores for the MouseTwo condition are significantly less than for other two conditions. Comparing the OneTwo and MouseOne conditions yield a probability that indicates similar means for these two conditions.

The Back scores for the MouseTwo condition are significantly less than the OneTwo score only. The other tests indicate that the MouseOne scores could be the same as either or both of the other two scores.

Table 7.3 presents tests for difference of End fatigue scores, with means briefly summarized. Again, the test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Fatigue End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Eyes/Face	-.58	.12	.56	.864	.722T	.391
Neck	.20	.08	.00	.668	.605	.584T
Shoulders	.57	-.06	1.11	.032	.591	.021
Back	-.03	.00	.44	.562T	.153	.293

Table 7.3: Midway fatigue comparisons between conditions for head and torso.

For the End fatigue scores, the Eyes/Face, Neck and Back scores are similar for all three conditions.

The Shoulder scores for the MouseTwo condition are significantly less than for other two conditions. Comparing the OneTwo and MouseOne conditions yield a probability that indicates similar means for these two conditions.

In summary, where the statistical tests indicate a difference between conditions, ranking in order of increasing scores is as follows:

1. MouseTwo
2. MouseOne
3. OneTwo

Hypothesis 13 is false for the Shoulders in both surveys after the start of the experiment, and false for the Back during the Midway survey. This means that the interface seems to have an effect on subject head and torso fatigue. However, analyzing all fatigue ratings and fatigue by condition, none of the means exceed 0 by a statistically significant margin, so it seems that head and torso fatigue increase only slightly.

7.2 Head and Torso Pain

Table 7.4 indicates head and torso pain ratings for all subjects, with Shapiro-Wilks normality probabilities.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.11. All of the means seem quite small, and none show any significant difference from 0.

Tables A.12 through A.17 indicate that for the MouseOne, MouseTwo and OneTwo conditions, the null hypothesis should be accepted for all of the means. Since none of tests give evidence against the null hypothesis, we can assume that hypotheses 10, 11, and 12 can be accepted.

Pain All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	.1840	.7471	.01	.1963	.9750	.01
Neck	.2407	.6401	.01	.0648	.6913	.01
Shoulders	.3519	1.0273	.01	.1728	1.1163	.01
Back	.1358	.7699	.01	.0846	.9450	.01

Table 7.4: All-subject pain statistics for head and torso.

For hypothesis 13, pairwise comparisons are again made. Tables A.18 and A.19 present tests for difference of Midway and End pain scores. In summary, there is no statistically significant difference between conditions for any of the pain ratings for the head and torso. Hypothesis 13 therefore holds for head and torso pain.

7.3 Left Arm Fatigue

Table 7.5 indicates left arm fatigue ratings for all subjects, with Shapiro-Wilks normality probabilities.

Fatigue All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.7222	1.6118	.01	.7346	1.2984	.01
Left Wrist	.5494	1.7352	.01	.5247	1.3735	.01
Left Arm	.3364	1.7176	.01	.2191	1.4445	.01

Table 7.5: All-subject fatigue statistics for the left arm.

Again, none of these means are normally distributed. The null hypothesis tests are in table 7.6.

Fatigue All	Midway=0	End=0
Region	2-tail P	2-tail P
Left Hand	.0382	.0218
Left Wrist	.0747	.0630
Left Arm	.4017	.6121

Table 7.6: Null hypothesis tests for all-subject left arm fatigue scores.

In this case, the left hand means both seem to differ significantly from zero. As the following analysis will show, this is partly because of a difference in means between conditions. The standard deviations for these scores are larger than for the head and torso fatigue, which also indicates a difference between conditions.

Analyzing by condition yields table 7.7 for the MouseOne case.

Testing for the null hypothesis yields the table 7.8.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table 7.9 is for the MouseTwo case.

Fatigue MouseOne	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.5796	1.6179	.01	.2463	.4633	.03
Left Wrist	.8519	2.4221	.01	.5556	1.6186	.01
Left Arm	.1000	.8042	.19	-.1556	.6481	.01

Table 7.7: Fatigue statistics for the left arm for the MouseOne condition.

Fatigue MouseOne	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Left Hand			.2850	.2850
Left Wrist			.1797	.1797
Left Arm	.847			.6547

Table 7.8: Testing null hypothesis for left arm fatigue in the MouseOne condition.

Testing for the null hypothesis yields table 7.10.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table 7.11 is for the OneTwo case.

Testing for the null hypothesis yields table 7.12.

For the left hand, the Midway and End scores are both normally distributed, and yield probabilities less than 0.10. The T-Test for the End score yields 0.01, a strongly significant result. The null hypothesis must therefore be rejected.

The 95% confidence interval for the left hand fatigue rating for the OneTwo condition is (.771..3.562), which barely includes 1 in the range. Taking the mean at face value, it would seem that the increase in fatigue for the left hand under this condition exceeds 1, falsifying hypothesis 11 for the left hand in this condition. Hypothesis 10 can only be tentatively accepted, because the null hypothesis is accepted by only a slight margin. Testing hypothesis 12 for this case with a paired-samples T-Test, the significance probability of a difference between the two means is 0.530, indicating that the Midway and End means are likely the same.

The tests for the left wrist and the left arm yield probabilities greater than 0.10, so the null hypothesis of means not different from zero should probably be accepted for these body regions.

For hypothesis 13, pairwise comparisons are again made.

Table 7.13 presents tests for difference of Midway fatigue scores, with means briefly summarized.

Fatigue MouseTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.0000	.6325	.20	.0000	.6325	.20
Left Wrist	-.1296	.4082	.01	.0000	.6325	.20
Left Arm	-.1296	.4082	.01	-.3148	.5164	.01

Table 7.9: Fatigue statistics for the left arm for the MouseTwo condition.

Fatigue MouseTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Left Hand	1.000	1.000		
Left Wrist		1.000	.3173	
Left Arm			.3173	.1797

Table 7.10: Testing null hypothesis for left arm fatigue in the MouseTwo condition.

Fatigue OneTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	1.6517	1.9408	.41	2.3423	1.3292	.54
Left Wrist	1.0412	1.6330	.08	1.0741	1.6021	.07
Left Arm	1.5741	2.5820	.01	1.4444	1.9748	.21

Table 7.11: Fatigue statistics for the left arm for the OneTwo condition.

The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Table 7.14 presents tests for difference of End fatigue scores, with means briefly summarized.

At the Midway point, there is a statistically significant difference between the MouseTwo case and the OneTwo case for the Left Wrist. At the End point, there are statistically significant differences between OneTwo and the other 2 cases for the Left Hand, and a between the MouseTwo case and the OneTwo case for the Left Arm. The fatigue rating for the Left Wrist is no longer significantly different.

In summary, where the statistical tests indicate a difference between conditions, ranking in order of increasing scores is as follows:

1. MouseTwo
2. MouseOne
3. OneTwo

Hypothesis 13 is false for the Left Wrist at the Midway point, and for the Left Hand and the Left Arm at the End survey. Combined with the previous tests comparing the ratings with 0, there is good evidence that the Left Hand experiences an increase in fatigue in the OneTwo case. There is moderate evidence that the Left Arm experiences an increase in fatigue in the OneTwo case, because the significance level of the difference from 0 is 0.122 for the Left Arm in the OneTwo case.

Fatigue OneTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Left Hand	.069	.010		
Left Wrist			.1088	.1088
Left Arm		.122	.1797	

Table 7.12: Testing null hypothesis for left arm fatigue in the OneTwo condition.

Fatigue Midway	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Left Hand	.58	.00	1.65	.592	.184	.070T
Left Wrist	.85	-.13	1.04	.093	.591	.045
Left Arm	.10	-.13	1.57	.391	.474	.092

Table 7.13: Midway fatigue comparisons between conditions for the left arm.

Fatigue End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Left Hand	.25	.00	2.34	.655	.017	.005T
Left Wrist	.56	.00	1.07	.341	.531	.128
Left Arm	-.16	-.32	1.44	.445	.107	.031

Table 7.14: End fatigue comparisons between conditions for the left arm.

7.4 Left Arm Pain

Table 7.15 indicates left arm pain ratings for all subjects, with Shapiro-Wilks normality probabilities.

Pain All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.4630	1.4605	.01	.3580	1.2423	.01
Left Wrist	.3457	1.5782	.01	.1728	1.0166	.01
Left Arm	.3642	1.6830	.01	.3210	1.0369	.01

Table 7.15: All-subject pain statistics for the left arm.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.31. All of these means are fairly small, and none is significantly different from 0.

Tables A.32 through A.37 indicate that none of the scores are statistically significantly different from 0, indicating that hypotheses 10, 11, and 12 can be accepted.

For hypothesis 13, pairwise comparisons indicate that there is no statistically significant difference between scores under any condition at both the Midway and End points. Hypothesis 13 holds for left arm pain, as shown in tables A.38 and A.39.

7.5 Right Arm Fatigue

Table 7.16 indicates right arm fatigue ratings for all subjects, with Shapiro-Wilks normality probabilities.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.41. In this case, none of the right arm means differ significantly from zero.

Tables A.42 through A.47 indicate that for the right arm as a whole, none of the scores are statistically significantly different from 0, indicating that hypotheses 10, 11, and 12 can be accepted.

For hypothesis 13, pairwise comparisons indicate that there is no statistically significant differ-

Fatigue All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.1687	1.2580	.01	.5123	1.1742	.01
Right Wrist	.2000	1.4389	.01	.3580	1.0604	.01
Right Arm	.0926	1.3959	.01	.1296	1.1540	.01

Table 7.16: All-subject fatigue statistics for the right arm.

ence between scores under any condition at both the Midway and End points. Hypothesis 13 holds for right arm fatigue, as shown in tables A.58 and A.59.

7.6 Right Arm Pain

Table 7.17 indicates right arm pain ratings for all subjects, with Shapiro-Wilks normality probabilities.

Pain All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.3519	1.0911	.01	.1543	.7078	.01
Right Wrist	.1605	.5469	.01	.0247	.3557	.01
Right Arm	.1049	.6853	.01	.0370	.5053	.01

Table 7.17: All-subject pain statistics for the right arm.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.51. All of these means are fairly small, and none is significantly different from 0.

Tables A.52 through A.57 indicate that for the arm as a whole, none of the scores are statistically significantly different from 0, indicating that hypotheses 10, 11, and 12 can be accepted.

For hypothesis 13, pairwise comparisons are again made.

Table 7.18 presents tests for difference of Midway pain scores, with means briefly summarized.

Pain Midway	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Right Hand	.08	.13	.00	.2518	.6074	.8485
Right Wrist	.40	.00	.26	.0578	.3900	.3173
Right Arm	.39	-.13	.13	.0449	.2830	.5982

Table 7.18: Midway pain comparisons between conditions for the right arm.

Table 7.19 presents tests for difference of End pain scores, with means briefly summarized.

At the Midway point, there is a statistically significant difference in Right Arm scores between the MouseOne and MouseTwo cases. However, the actual difference between the scores is very small, so it is likely that this is a chance occurrence. For the remainder, since there is no statistically significant difference between scores under any condition at both the Midway and End points, Hypothesis 13 holds for right arm pain.

Pain End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Right Hand	.33	.00	.13	.1396	.3900	1.0000
Right Wrist	-.08	.00	.13	1.0000	.5283	.3173
Right Arm	.24	.41	.00	.0927	.3900	.5982

Table 7.19: End pain comparisons between conditions for the right arm.

7.7 Pain and Fatigue Discussion

There are a number of general results of the pain and fatigue survey.

1. The only statistically significant increase in *pain* in any body part at any time for any condition is for the Right Wrist and Right Arm at the Midway point. However, the differences in actual values is so small that these differences can be ignored.

2. Midway pain and fatigue is usually higher than End pain and fatigue, indicating a habituation response.

3. The Left Hand shows a statistically significant increase in fatigue at the end of the trials for the OneTwo case. The increase in Left Hand fatigue means for all subjects is due to the increase in the OneTwo case.

4. When statistically significant differences occur between interfaces, the order of interfaces pairs is MouseTwo, MouseOne, OneTwo. At the Midway point, these differences occur in Left Wrist, Shoulders, and Back fatigue. At the End, Left Hand, Left Arm, and Shoulders fatigue are statistically significant between conditions.

One possible explanatory factor for a difference between conditions is that the subjects in the OneTwo condition are apt to feel fatigue, but this would result in consistently higher scores for all body parts. Such is not the case, as illustrated by the Eyes/Face and Neck fatigue scores.

A superficial interpretation of these results is that the mere combination of the One-Handed and Two-Handed interfaces is to blame for left arm fatigue. This would imply that the One-Handed and Two-Handed two styles should not be combined. However, if differing hand roles alone were to blame, then we might expect to see higher fatigue right arm ratings for the conditions that include the mouse, and we do not.

The surprise of this survey is that there is a difference in overall left arm fatigue, especially considering that the MouseTwo case consistently has at or near the lowest scores. At first blush, it would seem that the left arm is idle for the Mouse and the One-Handed interfaces, but this is not in fact the case.

While using the Mouse-based interface, the subject's left hand is idle, so the subject can rest it if necessary. Consequently, fatigue scores for conditions that include the Mouse-based interface are the lowest, because the left arm is idle for half of the experiment.

In the Two-Handed interface, the subject manipulates the left bat. Fatigue scores in the MouseTwo condition are the lowest, while the highest scores occur in the OneTwo condition.

In the One-Handed interface, the subject selects keyboard commands with the left hand from the 1, 2 and 3 keys. Fatigue scores for conditions that include the One-Handed interface are the highest.

Taken together, this indicates that the higher fatigue scores for the left arm in the OneTwo case mostly arise from the use of the One-Handed interface. This in turn indicates that the use of the keyboard as a substitute for buttons on the bat is probably a bad idea. In a review of various types of computer-related workplace injury [57], Sellers notes that the leading cause of Cumulative Trauma Disorders (CTD) such as Carpal Tunnel Syndrome is static posture. He reports that the accepted recommendation for avoiding such problems is to frequently change postures. The One-Handed interface does not easily offer this opportunity, because the left hand is poised over the 1, 2 and 3 keys for the duration of each trial.

To add evidence to this view, the Shoulder fatigue scores are also highest when the One-Handed interface is used, and is lowest for the MouseTwo case. In fact, comparing between conditions indicates that the MouseTwo Shoulders scores are significantly lower from the other two, and that the MouseOne and OneTwo scores do not differ from each other. Sellers mentions that CTD symptoms are not confined to the hands, but can be displaced into the shoulders and neck area. This is consistent with idea that left arm fatigue arises from the static posture required by the One-Handed interface.

Chapter 8

Experimental Results – Preferences

This chapter explains the results of the familiarity and preferences survey. Analyzing the preference data survey is a little simpler than the Pain and Fatigue survey, because each of the ratings questions are clear about what is being compared.

A total of 123 familiarity ratings were collected:

$$(4\ \textit{Page}\ 5\ \textit{Questions} + 3\ \textit{Page}\ 9\ \textit{Questions}) \times\ 18\ \textit{Subjects}\ -\ 3\ \textit{Missing}$$

For the general preference questions, a total of 180 ratings were collected:

$$5\ \textit{Questions}\ \times\ 2\ \textit{Surveys}\ \times\ 18\ \textit{Subjects}$$

For the detailed preference questions, a total of 315 ratings were collected. The missing data occurred because the subject did not notice that the order of the rating scales had changed in the End survey.

$$9\ \textit{Questions}\ \times\ 2\ \textit{Surveys}\ \times\ 18\ \textit{Subjects}\ -\ 9\ \textit{Missing}$$

Like the pain and fatigue survey, the questions that compare interface A to interface B occur in blocks of 6 subjects each. However, appropriate pairs of questions can be combined into a group of 12 responses. For example, the Mouse vs. OneHanded question and the Mouse vs. TwoHanded question each have 6 responses. Reinterpreting the meaning of the rating to be Mouse vs non-Mouse, these two questions can be combined yielding 12 responses. This helps the analysis by narrowing the 95% confidence interval for this type of question.

8.1 Familiarity

Hypothesis 14: There is no difference in familiarity ratings between conditions.

Hypothesis 15: Familiarity ratings for One-Handed and Two-Handed interfaces are less than for the Mouse-based interface at the Midway point.

Table 8.1 lists the means, standard deviations and Shapiro-Wilks tests for normality for the familiarity survey. Again, the Tukey Biweight M-Estimator is used to report central tendency instead of the arithmetic mean.

Familiarity All	Midway			End		
Topic	M-Est	StDev	S-W	M-Est	StDev	S-W
AutoCAD	4.8405	2.1404	.33			
Grid Creation	6.0790	2.8939	.23			
Box Folding				6.4042	2.6111	.26
Mouse	6.7855	2.4051	.74	6.8498	2.7609	.47
OneHanded	1.8045	2.7494	.04	5.3721	2.2508	.96
TwoHanded	3.8196	3.1130	.56	5.6456	2.6799	.48

Table 8.1: All-subject familiarity ratings. The interface ratings have 12 responses each, while the task and AutoCAD ratings have 18.

These responses confirm the results of the training and practice time analysis from section 6.2.8. At the Midway point, the Mouse-based interface is rated at 6.8, indicating higher than moderate familiarity and expertise, while the One-Handed interface rates 1.8, and the Two-Handed interface rates 3.8. AutoCAD is actually rated lower than the Mouse-based interface, at 4.8, and this is likely because subjects are accounting for AutoCAD's much larger command set.

Breaking up the midway ratings by condition yields table 8.2.

Midway	MouseOne			MouseTwo			OneTwo		
Topic	Mean	StDv	SW	Mean	StDv	SW	Mean	StDv	SW
AutoCAD	5.12	2.56	.76	3.25	2.17	.59	5.55	.98	.02
Grid	6.50	2.17	.66	4.64	3.65	.53	7.10	3.06	.07
Mouse	6.01	2.82	.93	7.74	1.82	.10			
OneHanded	1.31	3.35	.04				3.86	2.28	.06
TwoHanded				3.97	4.06	.16	3.40	2.48	.74

Table 8.2: All-subject midway familiarity ratings displayed by condition.

Most of these conditions seem to have normal distributions except the OneHanded familiarity ratings, and the AutoCAD and Grid ratings for the OneTwo condition. To test hypothesis 14, an analysis of variance in the means for each familiarity rating is shown in table 8.3. The One-Handed case uses a Wilcoxon test, while the remainder use an F test. The table shows that there is no significant difference between conditions for any Midway familiarity rating.

To test hypothesis 15, the difference in familiarity means between conditions must be analyzed. Table 8.4 shows the results of paired-samples T-Tests for the Mouse, One-Handed and Two-Handed ratings. Unlike the previous tests, these comparisons are within-subject, so the problem of each subject using an independent incompatible scale does not exist. The probabilities are quite close to .05, indicating that the Mouse-based system is likely to be more familiar than the One-Handed or Two-Handed interface.

Breaking up the End ratings by condition yields table 8.5. By the end of the experiment, the

Topic	Number of Conditions	F Probability
AutoCAD	3	.1653
Grid	3	.5654
Mouse	2	.3817
OneHanded	2	.3269W
TwoHanded	2	.9350

Table 8.3: All-subject analysis of variance by condition for midway familiarity.

Pairwise Midway	Significance	
Interface	Mouse	OneHanded
OneHanded	.077	
TwoHanded	.067	.741

Table 8.4: Pairwise comparisons of midway interface familiarity ratings.

ratings have moved towards the middle zone, most lying in the range of 4-6.5.

End	MouseOne			MouseTwo			OneTwo		
Topic	Mean	StDv	SW	Mean	StDv	SW	Mean	StDv	SW
Box	6.48	2.42	.32	8.20	3.33	.02	6.00	2.25	.47
Mouse	7.21	2.18	.41	5.79	3.20	.72			
OneHanded	4.00	2.83	.42				6.27	1.47	.71
TwoHanded				4.87	3.31	.77	6.36	1.95	.41

Table 8.5: All-subject End familiarity ratings displayed by condition.

To test hypothesis 14, an analysis of variance in the means for each End familiarity rating is shown in table 8.6. The table shows that there is no significant difference between conditions for any End familiarity rating.

Hypothesis 16: Familiarity ratings for all interfaces do not change from the Midway point to the End point.

Table 8.7 shows the results of paired-samples T-Tests for the Mouse, One-Handed and Two-Handed ratings. Again, these comparisons are within-subject, so between-subject variability is accounted for. Only the OneHanded interface still seems unfamiliar compared to the Mouse.

8.2 Familiarity Discussion

The main point of this analysis is that the subjects' self-rating of familiarity and expertise with the interfaces matches the results obtained from the training and practice timings. The longer training and practice timings occur in unfamiliar interfaces.

Another interesting result is that subjects do not gain as much familiarity with the One-Handed interface as with the Two-Handed interface. The MouseOne condition yields a difference in familiarity ratings that does not seem to close as quickly as the MouseTwo condition.

Topic	Number of Conditions	F Probability
Box	3	.6776
Mouse	2	.3031
OneHanded	2	.3605
TwoHanded	2	.4148

Table 8.6: All-subject analysis of variance by condition for End familiarity.

Pairwise End	Significance	
Interface	Mouse	OneHanded
OneHanded	.110	
TwoHanded	.363	1.000

Table 8.7: Pairwise comparisons of End interface familiarity ratings.

8.3 General Preferences

Table 8.8 lists the means, standard deviations and Shapiro-Wilks tests for normality for the preference survey questions that asked for a general interface rating. Each question asked the subject to give a rating from 0 to 10 for each of the two interfaces being used. There is no explicit comparison being asked for in these two scales, so these are referred to as *single-ended* ratings. Double-ended ratings are also used in the preference survey, and these explicitly compare Interface A to Interface B on a scale from -5 to 5. Again, the Tukey Biweight M-Estimator is used in this table to report central tendency instead of the arithmetic mean.

Preferences All		Midway			End		
Interface	Condition	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	MouseOne	7.6280	1.8552	.86	7.5000	2.5720	.28
Mouse	MouseTwo	7.7593	1.3292	.01	7.1047	1.1690	.39
OneHanded	MouseOne	6.1619	1.8115	.15	6.6819	2.3583	.40
OneHanded	OneTwo	6.9259	1.6021	.07	6.3456	2.0656	.40
TwoHanded	MouseTwo	8.0000	.8944	.13	8.1012	1.7224	.82
TwoHanded	OneTwo	6.8148	1.7224	.04	7.0815	1.0328	.48

Table 8.8: All-subject single-ended preference ratings. Each of these ratings has 6 responses each on a scale from 0 to 10.

There doesn't seem to be much difference in preference ratings between conditions for each interface except for the Two-Handed interface. In particular, something quite interesting happens in the One-Handed and Two-Handed ratings for the Midway OneTwo condition, as shown in table 8.9. The last entry includes the double-ended rating that explicitly compares the interfaces to each other, with positive values in this case indicating a preference for the Two-Handed interface, and negative values indicating One-Handed preference. The table clearly shows that the ratings are directly related to presentation order. That is, by a fairly wide margin, subjects prefer the second interface that was presented. However, since presentation order was balanced, this split has no deleterious effect on the aggregate statistics. This bimodal distribution does not continue into the detailed

comparisons of interface elements, nor does it continue to the End ratings.

OneTwo Preferences	OneHanded First		TwoHanded First	
Interface	M-Est	StDev	M-Est	StDev
OneHanded	5.7562	1.5275	8.0000	.0000
TwoHanded	8.0000	1.7321	5.6666	.5774
One vs Two	4.3333	.5774	-3.0000	1.7321

Table 8.9: Preference ratings for One-Handed and Two-Handed interfaces in the OneTwo condition accounting for presentation order.

Hypothesis 17: There is no difference between conditions for the general single-ended preference ratings for any interface.

To examine whether the experimental condition affect the single-ended ratings given to an interface, table 8.10 shows the results of statistical tests of difference between the component means for each interface rated by condition. Since there is some question of normality in the Midway means, a Mann-Whitney U test will be used for these tests, while an independent samples T-Test will be used for the End means. Table 8.10 also reports the aggregate means.

Bulk Preferences	Midway			End		
Interface	M-Est	StDev	M-W Sig	M-Est	StDev	T-Test Sig
Mouse	7.6445	1.5442	1.0000	7.3566	1.9057	.9214
OneHanded	6.7406	1.6728	.4507	6.3482	2.1167	.8689
TwoHanded	7.4436	1.4434	.1874	7.5090	1.3790	.5556

Table 8.10: All-subject single-ended preference ratings aggregated from all conditions, and tests of differences between means. Each of these ratings has 12 responses each on a scale from 0 to 10.

Considering the size of the standard deviations, there is no point in testing for a difference in means between each interface type. The general trend of the numbers suggests that the Mouse and Two-Handed interfaces are generally preferred over the One-Handed interface, and that this trend widens at the end.

8.4 General Comparative Preferences

The first two questions in the preference survey asked for single-ended ratings. The remaining questions ask for a double-ended comparison between systems. This section will deal with the questions that ask for a general comparison, while section 8.6 deals with individual interface components.

For each of these ratings, the subject was asked to rate on a scale from -5 to 5, with -5 indicating strong preference for one interface, 0 indicating neutrality, and 5 indicating preference for the other interface. Because there are three interfaces, a single numeric scale cannot be used to summarize collections of pairwise preference data, so the following tables must be read carefully to avoid misinterpretation. For example, a positive value in one question may have the same meaning as a negative value in another. In table 8.11, the column headed *negative* indicates the interface

associated with negative values, and the column headed *positive* indicates the interface associated with positive values. Also, although the assignment of positive and negative values to interfaces was swapped in the End questionnaire from the Midway questionnaire's order, the values have been recoded into the Midway scheme to avoid confusion. This recoding was unnecessary for the AutoCAD comparisons, because AutoCAD was always assigned to the negative end of the scale. Table 8.11 displays these ratings, with two lines for each of the AutoCAD comparisons corresponding to the two experimental conditions that the given interface was compared in.

Comparisons			Midway			End		
Negative	Positive	Condn	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One		-.0926	3.1791	.98	.2340	2.6394	.18
Mouse	Two		1.5354	2.3752	.48	1.2973	2.2249	.52
One	Two		.8240	4.1793	.16	1.0234	2.8107	.05
AutoCAD	Mouse	M1	2.1890	1.2813	.11	2.0242	2.9439	.45
AutoCAD	Mouse	M2	1.1916	2.4290	.23	1.9983	2.6077	.05
AutoCAD	One	M1	1.5985	1.9937	.85	1.4755	1.1225	.09
AutoCAD	One	12	2.3120	2.9439	.21	1.3420	2.0656	.40
AutoCAD	Two	M2	2.7116	2.4833	.05	2.9993	3.1885	.36
AutoCAD	Two	12	2.1151	1.8348	.49	2.7951	2.7142	.09

Table 8.11: All-subject double-ended comparisons. A negative rating indicates preference for the interface in the column labeled *Negative*. Positive ratings correspond to the *Positive* column interface. Each of these means has 6 responses each on a scale from -5 to 5.

8.4.1 AutoCAD Comparative Preferences

Since each subject compares 2 experimental interfaces to AutoCAD, there are a total of 12 comparisons of AutoCAD to each interface, with 6 ratings per appropriate experimental condition. It makes sense to aggregate each of these 2 groups of 6 ratings as long as there is no difference between conditions.

Hypothesis 18: AutoCAD comparisons are not affected by which pair of interfaces are being used.

Table 8.12 shows the aggregate means of the AutoCAD comparisons and their standard deviations. The significance columns are probabilities that the two components that make up each aggregate mean have different means. Most of the Midway component ratings were normally distributed, so the significance probabilities ending in T denote an independent samples T-Test. The Midway Two-Handed significance (labeled M) is a Mann-Whitney U test, as are all of the End tests.

The low significance probability of the difference in means between the two AutoCAD versus Mouse comparisons indicates that the experimental condition might be having an effect on the scores, contrary to hypothesis 18. In the MouseOne condition, the two AutoCAD comparisons differ from each other by 0.6 in the Mouse-based interface's favor, while in the MouseTwo condition, the scores differ by 1.5 in favor of the Two-Handed interface. For the remaining AutoCAD comparisons, the significance probabilities indicate that the experimental condition has no effect.

Hypothesis 19: AutoCAD comparisons prefer the experimental interface over AutoCAD.

AutoCAD vs:	Midway			End		
Interface	M-Est	StDev	Sig	M-Est	StDev	Sig
Mouse	2.1506	2.1047	.118T	2.0395	2.6959	.626M
OneHanded	1.7693	2.3975	.955T	1.7616	1.6124	.684M
TwoHanded	2.5410	2.0817	.806M	3.0679	2.8284	.806M

Table 8.12: All-subject AutoCAD comparison ratings aggregated from two conditions per aggregate, with tests of differences between component means across the two conditions. Each of these aggregate ratings has 12 responses on a scale from -5 to 5.

Table 8.13 briefly summarizes the aggregate comparisons of AutoCAD versus the experimental interfaces in increasing order of preference. The S-W columns are the Shapiro-Wilks tests of normality for the aggregate scores, and the Sig columns indicate 1-tailed significances for hypothesis 19. Significances labeled T are T-Tests, while W indicates Wilcoxon tests. The significance results indicate quite strongly that the AutoCAD is the least preferred interface. The scores indicate again that the Two-Handed system is the most preferred, followed by the Mouse-based system, the One-Handed system and lastly, AutoCAD. Also, the End score for the TwoHanded system increases over the Midway score.

AutoCAD Rankings	Midway			End		
Interface	M-Est	S-W	Sig	M-Est	S-W	Sig
OneHanded	1.8	.11	.045T	1.8	.37	.002T
Mouse	2.2	.07	.033W	2.0	.41	.043T
TwoHanded	2.5	.05	.011W	3.1	.02	.016W

Table 8.13: Aggregate rankings of interfaces versus AutoCAD. The significance column contains 1-tailed probabilities that the mean ranks are greater than zero, indicating a preference of the given interface over AutoCAD. Each of these aggregate ratings has 12 responses on a scale from -5 to 5.

8.4.2 Experimental Interface Comparative Preferences

Hypothesis 20: The Two-Handed Interface is preferred over the other experimental interfaces.

The last set of general comparisons to consider are those between Interface A and Interface B. There are three pairs of double-ended ratings, reprised in table 8.14, which also shows statistical tests for the null hypothesis. Significance probabilities suffixed with T are T-Tests, while W indicates Wilcoxon tests. None of these significances indicate strong evidence against the null hypothesis, although the general trend again indicates that the Two-Handed system is the most preferred.

As mentioned early in the chapter, appropriate pairs of ratings with 6 responses can be aggregated into a group of 12 as long as the meaning of the ratings is appropriately reinterpreted. Therefore, the three new aggregate scores are entitled *NotMouse vs. Mouse*, which takes the Mouse vs. One-Handed and Mouse vs. Two-Handed double-ended scores. Similarly, the *NotOne-Handed vs. One-Handed*, and *NotTwo-Handed vs. Two-Handed* aggregates take the appropriate double-ended scores. Unlike the AutoCAD ratings, the arithmetic of aggregation for these scores must take into account what end of the scale was assigned to what interface. Positive values of an aggregate

Comparisons		Midway		End	
Negative	Positive	M-Est	Sig	M-Est	Sig
Mouse	One	-.09	.922T	.23	.747T
Mouse	Two	1.54	.315T	1.30	.261T
One	Two	.82	.712T	1.02	.345W

Table 8.14: General double-ended comparisons with null hypothesis tests. Each of these ratings has 6 responses each on a scale from -5 to 5.

score are assigned to the positive statement of the interface, and negatives are assigned to the *Not* case, for example *NotMouse*.

Table 8.15 shows these aggregate scores. Like the AutoCAD scores in table 8.12, there is a concern that the component means are statistically similar, so the Sig columns list the results of independent-samples statistical tests for equality of means. The significance probabilities suffixed with T are T-Tests results, while M indicates Mann-Whitney U tests due to non-normality of the End One-Handed vs. Two-Handed component.

NotInterface vs:	Midway			End		
Interface	M-Est	StDev	Sig	M-Est	StDev	Sig
Mouse	-.7183	2.7499	.470T	-.8332	2.2249	.188T
OneHanded	-.3894	3.5512	.809T	-.5166	2.6661	.292M
TwoHanded	1.1204	3.2483	.836T	1.3670	2.4374	.853M

Table 8.15: Aggregated comparisons between Interface and Not-Interface. Each of these aggregate ratings has 12 responses on a scale from -5 to 5, with negative values indicating preference against the given interface.

The results show that only the Two-Handed interface has a positive score, with the Mouse-based interface preferred the least.

Table 8.16 briefly summarizes the aggregate interface rankings, with Shapiro-Wilks tests of normality. The Sig columns indicate 2-tailed significances for the null hypothesis. Significances suffixed with T are T-Tests, while W indicates Wilcoxon tests. For the Midway scores, the significance probabilities fail to reject the null hypothesis. For the End Two-Handed score, the 2-tailed probability of .084 is small, indicating a real user preference for the Two-Handed interface. This implies that the 91% confidence interval for the NotTwo-Handed vs. Two-Handed score is (2.789, .029). Testing hypothesis 20 yields a 1-tailed probability of .042, which is within the acceptance range for significance.

8.5 General Comparison Discussion

The central message of all the general comparison results is that the subjects preferred the Two-Handed interface to all other contenders. This preference was not unanimous, since for example, two subjects favored AutoCAD over the Two-Handed interface in the End survey. Part of the intent of the AutoCAD question was to encourage the subject to consider the idea of abandoning their 3 months investment in AutoCAD in favor of the experimental system. Two were unwilling to do so,

Interface Rankings	Midway			End		
Interface	M-Est	S-W	Sig	M-Est	S-W	Sig
Mouse	-.7	.62	.562T	-.8	.75	.615T
OneHanded	-.4	.33	.704T	-.5	.06	.359W
TwoHanded	1.1	.37	.371T	1.4	.20	.084T

Table 8.16: Interface vs NotInterface with null hypothesis tests. The T suffix indicates a T-Test, while W indicates a Wilcoxon test.

although their preferences may have arisen from AutoCAD's richer command set.

One other issue that is hopefully handled by pairwise comparisons is the *halo effect*. That is, because the subject wants to please the experimenter and "do well" in the experiment, the subject tells the experimenter what he/she wants to hear. To do this, the subject must figure out what the experimental goal is, and with new technologies, the goal is fairly obvious – it's to prove that the new technology is spiffier than the current technology. Thus, the halo effect comes into play when the new technology acquires a more positive light than it deserves by dint of its newness.

My solution to this was to present three new technologies, and to present them without bias or hype. Randomizing and balancing presentation order helps make guessing the thesis more difficult, as does proper conduct during the experiment. For example, during the trials, some subjects asked me what my thesis was about, or whether I had written the programs they were using. My response to these questions was to change the subject or, failing that, to say that I couldn't answer the question.

One other event related to this occurred when one of the subjects in the OneTwo condition was being trained on the One-Handed interface after having completed the first set of Two-Handed trials. Upon hearing that all of the spatial operations had moved onto the right bat, the subject said that the One-Handed scheme seemed more logical to him. The above analysis has simply included his survey responses, but if we consider him as a biased observer (since he expressed his opinion before seeing the system), the results for the Midway questions move towards greater preference for the Two-Handed interface from 1.1 to 1.6, and the standard deviation drops by 0.45. In fact, at no point did he express a positive opinion about the Two-Handed system in the comparative ratings. This bias reflects itself in strong One-Handed preferences for some of the following detailed preferences.

The End survey results don't change all that much, because this subject moderated his opinion somewhat, and ended by preferring neither interface over the other in general.

Again, the lack of unanimity in the survey data indicates that subjects were probably making real choices, and were spending only slight energy trying to figure out what the experimenter wanted to hear. Even the biased subject thought carefully about the survey questions, and the mere *presence* of a subject biased against the Two-Handed interface lends credence to the idea that the procedures for guarding against experimenter-generated bias were working.

8.6 Detailed Comparative Preferences

The foregoing discussion relates to the question of a summative evaluation of the interface that the subject knows about. General preferences don't really say much about the details of the interface, they just evaluate whether the interfaces work well as a whole. However, each interface *is* made up of parts, so this section analyzes the questions asked about detailed interface components.

Like the general comparisons, the encoding of the detailed preferences is a double-ended scale, with positive values assigned to one interface, and negative values assigned to the other. Also, the Midway assignment was opposite to the End assignment, and the assignment was based on the presentation order, so no interface was consistently assigned to one end of the scale.

The question that asked the subject to compare interfaces for *Moving Without Constraints* has been left out of the following summaries because neither of the tasks required the subject to move a vertex without constraints. Most subjects left the question blank, or rated 0 for no preference.

8.6.1 Vertex Selection Preference

Hypothesis 21: There will be no difference between the three interfaces for vertex selection.

Table 8.17 summarizes the means, standard deviations and Shapiro-Wilks normality tests for the preference for Vertex Selection.

Vertex Selection		Midway			End		
Negative	Positive	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One	-1.3913	2.8472	.06	-.5358	2.9735	.48
Mouse	Two	-.1314	3.5024	.48	-.9976	2.4900	.29
One	Two	.0741	1.4720	.03	1.1284	1.3038	.24

Table 8.17: All-subject double-ended comparisons of vertex selection. A negative rating indicates preference for the interface in the column labeled *Negative*. Positive ratings correspond to the *Positive* column interface. Each of these means has 6 responses each on a scale from -5 to 5.

Table 8.18 shows tests for the null hypothesis for each of the 6 ratings. T-Tests are used when the ratings seem normally distributed, otherwise Wilcoxon tests are used. None of these indicate a statistically significant result, indicating that hypothesis 21 seems to hold. However, there seems to be a trend towards preferring the Mouse-based selection mechanism. There is also a preference for the Two-Handed vertex selection over the One-Handed version at the End, and this preference has the lowest significance probability. This is curious, since they operate identically in isolation from other interface elements. One possibility is that users feel more comfortable picking in the 3D task with two hands than with one.

8.6.2 Moving With Constraints Preference

Hypothesis 22: The Two-Handed interface will be preferred over the other two interfaces.

Table 8.19 summarizes the means, standard deviations and Shapiro-Wilks normality tests for the preference for Moving With Constraints.

Table 8.20 shows tests for the null hypothesis for each of the moving with constraints ratings.

Vertex Selection	T-Test Significance		Wilcoxon Significance	
Interfaces	Midway=0	End=0	Midway=0	End=0
MouseOne		.651	.3454	
MouseTwo	.825	.512		
OneTwo		.109	.3173	

Table 8.18: Null hypothesis tests for vertex selection preference.

Moving With Constraints		Midway			End		
Negative	Positive	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One	-2.6813	4.1104	.22	-.7354	2.4014	.31
Mouse	Two	1.1674	2.9269	.34	1.1111	2.8284	.02
One	Two	2.3615	1.5166	.49	2.2870	1.8166	.92

Table 8.19: All-subject double-ended comparisons of moving with constraints. A negative rating indicates preference for the interface in the column labeled *Negative*.

There is a statistically significant preference for the Two-Handed system over the One-Handed system in both the Midway and End surveys. In Mouse-based comparisons, the Two-Handed system is preferred and the One-Handed system is disliked. The Mouse-based system ranks in the middle. Hypothesis 22 can be accepted with respect to the One-Handed versus Two-Handed comparison, but there seems not to be definite evidence for a preference over the Mouse-based interface.

Moving With Constraints	T-Test Significance		Wilcoxon Significance	
Interfaces	Midway=0	End=0	Midway=0	End=0
MouseOne	.596	.287		
MouseTwo	.895			.4185
OneTwo	.019	.042		

Table 8.20: Null hypothesis tests for moving with constraints preference.

The reason that users disliked the One-Handed system is because they sometimes had to twist their right hand awkwardly to match the desired constraint axis, thus making it difficult to hit. In this situation, the user wants to orient the bat so that it is momentarily aligned correctly, then press button 2 to commence reshaping. Unfortunately, pressing the button will sometimes serve to move the bat out of alignment, causing a reshaping error. Putting some hysteresis in the alignment detector would reduce this effect. Probably a better syntax for the One-Handed system is to have constrained vertex movement occur in two steps, perhaps by stating the constraint axis first, then moving the vertex.

For the Two-Handed interface, using the left hand to independently state the constraint axis seems to be a successful technique. Some people require a little effort to find the desired axis, but one of the subjects said in the post-experiment interview that this was because she never looked at the left cursor. Clearly, this is one of the drawbacks of having two cursors, since the user must be able to visually locate both of them. In any case, some alignment hysteresis would help here, also.

8.6.3 Activating Rulers Preference

Hypothesis 23: There will be no difference between the three interfaces for activating rulers.

Table 8.21 summarizes the means, standard deviations and Shapiro-Wilks normality tests for the preference for Activating Rulers.

Activating Rulers		Midway			End		
Negative	Positive	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One	-2.2270	2.4983	.30	-1.9638	1.5832	.46
Mouse	Two	-1.5072	3.2506	.67	-.1111	2.3452	.20
One	Two	.7010	3.0111	.18	1.6168	2.3381	.39

Table 8.21: All-subject double-ended comparisons of activating rulers. A negative rating indicates preference for the interface in the column labeled *Negative*.

Table 8.22 shows tests for the null hypothesis for each of the activating rulers ratings. The MouseOne case shows a statistically significant-preference for the Mouse-based interface in the End survey, and an almost significant result in the Midway survey. None of the remaining tests indicate a statistically significant result, but there is a trend towards preference of the Mouse-based interface over the Two-Handed interface, and for the Two-Handed interface over the One-Handed interface. Hypothesis 23 does not hold for the MouseOne case.

Activating rulers is a compound task that requires vertex selection and menu selection with a child menu item. Both of these component tasks are independently rated in this questionnaire – vertex selection is rated two questions earlier, and menu selection is rated at the end. For MouseOne and OneTwo End conditions, these two component ratings *sum* to be approximately equal to the End ratings for this question. The MouseOne and OneTwo Midway ratings are approximately equal the *mean* of the two components under the same condition. For this question, the MouseTwo condition gives ratings that are somewhat less than the mean of the two corresponding components.

Activating Rulers	T-Test Significance	
Interfaces	Midway=0	End=0
MouseOne	.085	.017
MouseTwo	.420	1.000
OneTwo	.611	.221

Table 8.22: Null hypothesis T-Tests for activating rulers preference.

8.6.4 View Control Preference

Hypothesis 24: The One-Handed and Two-Handed interfaces will be preferred over the Mouse-based system for view control.

Table 8.23 summarizes the means, standard deviations and Shapiro-Wilks normality tests for the preference for View Control.

Table 8.24 shows tests for the null hypothesis for each of the view control ratings. Preference is strongly against the Mouse-based interface, with a statistically significant preference for the One-

View Control		Midway			End		
Negative	Positive	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One	1.1982	4.0373	.31	3.5037	.9832	.01
Mouse	Two	3.3463	3.6009	.01	3.0000	3.2711	.02
One	Two	3.8878	1.9235	.33	3.1867	3.6697	.19

Table 8.23: All-subject double-ended comparisons of view control. A negative rating indicates preference for the interface in the column labeled *Negative*.

Handed interface at the End. The Midway OneTwo rating gives a statistically significant preference for the Two-Handed interface. The MouseTwo rating has equally large means, but less complete consensus about preference. These results verify Ware's results [75], where the bat was preferred over the mouse for examining 3D objects. Hypothesis 24 can be accepted for the MouseOne case, but there is a surprising preference of the Two-Handed interface over the One-Handed interface.

This strong preference is not too surprising, since most people consider that free examination of the 3D scene is one of the tasks that the Bat is really good for. The clear preference for the Two-Handed over the One-Handed interface probably stems from the use of the left hand for this task. One subject commented that the Two-Handed system gave him an enhanced sense of the space he was working in, while the One-Handed system did not.

View Control	T-Test Significance		Wilcoxon Significance	
Interfaces	Midway=0	End=0	Midway=0	End=0
MouseOne	.609			.0277
MouseTwo			.2945	.4185
OneTwo	.020	.317		

Table 8.24: Null hypothesis tests for view control preference.

8.6.5 Examining Surface Preference

Hypothesis 25: The One-Handed and Two-Handed interfaces will be preferred over the Mouse-based system for examining the surface.

Table 8.25 summarizes the means, standard deviations and Shapiro-Wilks normality tests for the preference for Examining the Surface.

Examining Surface		Midway			End		
Negative	Positive	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One	2.9864	3.8264	.32	3.1154	2.6230	.16
Mouse	Two	3.1629	3.1252	.28	3.7843	.8367	.25
One	Two	4.2883	2.7019	.11	3.7218	2.7019	.24

Table 8.25: All-subject double-ended comparisons of examining surface. A negative rating indicates preference for the interface in the column labeled *Negative*.

Table 8.26 shows tests for the null hypothesis for each of the examining surface ratings. There is

a statistically significant preference for the Two-Handed interface over the Mouse-based interface, and a nearly significant preference for the Two-Handed over the One-Handed interface at the End. The preference means for the MouseOne condition also indicate One-Handed preference. Hypothesis 25 can be accepted for the MouseTwo case, but there is again a surprising preference of the Two-Handed interface over the One-Handed interface.

This question is quite similar to the previous one, but there is a difference in results for the MouseTwo case that indicates that while the subjects like the general ability to turn the object over in real-time, some feel that the control of it could stand improvement.

Examining Surface	T-Test Significance	
Interfaces	Midway=0	End=0
MouseOne	.406	.195
MouseTwo	.210	.001
OneTwo	.118	.098

Table 8.26: Null hypothesis T-Tests for examining surface preference.

8.6.6 Examining Rulers Preference

Hypothesis 26: The One-Handed and Two-Handed interfaces will be preferred over the Mouse-based system for examining rulers.

Table 8.27 summarizes the means, standard deviations and Shapiro-Wilks normality tests for the preference for Examining Rulers.

Examining Rulers		Midway			End		
Negative	Positive	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One	1.5675	3.4412	.31	.6459	3.9021	.16
Mouse	Two	.2422	3.7103	.86	.8283	2.7749	.47
One	Two	4.2883	2.7019	.11	1.2990	3.0111	.18

Table 8.27: All-subject double-ended comparisons of examining rulers. A negative rating indicates preference for the interface in the column labeled *Negative*.

Table 8.28 shows tests for the null hypothesis for each of the examining rulers ratings. There are no statistically significant results for this question, arising from a slight preference against the Mouse-based system in favor of the One and Two-Handed interfaces. The large preferences in the Midway ratings in the conditions that include the One-Handed system do not remain at the End, indicating that the End ratings are probably more reliable indicators. There is no statistical evidence to accept hypothesis 26.

This question is also somewhat redundant, since examining rulers must be done by examining the surface, so the change in means probably indicates that the subjects are treating this question rather lightly.

Examining Rulers	T-Test Significance	
Interfaces	Midway=0	End=0
MouseOne	.543	.874
MouseTwo	.917	.554
OneTwo	.118	.328

Table 8.28: Null hypothesis T-Tests for examining rulers preference.

8.6.7 Menu Appearance Preference

Hypothesis 27: There will be no preference for any interface in menu appearance.

Table 8.29 summarizes the means, standard deviations and Shapiro-Wilks normality tests for the preference for Menu Appearance.

Menu Appearance		Midway			End		
Negative	Positive	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One	-1.2202	2.8592	.60	-1.1979	2.1012	.46
Mouse	Two	1.4924	3.2660	.41	1.2583	3.1937	.56
One	Two	2.5000	1.9494	.06	.2235	2.0000	.09

Table 8.29: All-subject double-ended comparisons of menu appearance. A negative rating indicates preference for the interface in the column labeled *Negative*.

Table 8.30 shows tests for the null hypothesis for each of the menu appearance ratings. There is no significant result for this question, although the Midway comparison for the OneTwo case and the End MouseOne comparison have low probabilities. Hypothesis 27 can probably be accepted. The trend seems to be in favor of the Mouse-based system in comparison to the One-Handed system. The numbers favoring the Two-Handed system over the Mouse-based system are not significantly different from 0. Like the previous question, the Midway number favoring Two-Handed over One-Handed shrinks somewhat at the end, indicating that the subject has considered the rating more carefully.

The purpose of this question was to ask the subject to rate the visual appeal of the sundial menus, and since the sundial is the same in the One and Two-Handed interface, the mean preference should be 0. The drop in preference in the OneTwo case indicates that the subject understands at the End what this question means.

Considering the MouseOne and MouseTwo conditions together, there opinion seems to be neutral about the look of the Sundial menus.

Menu Appearance	T-Test Significance		Wilcoxon Significance	
Interfaces	Midway=0	End=0	Midway=0	End=0
MouseOne	.333	.131		
MouseTwo	.363	.448		
OneTwo			.0796	.2850

Table 8.30: Null hypothesis T-Tests for menu appearance preference.

8.6.8 Menu Selection Preference

Hypothesis 28: *The Mouse-based interface will be preferred for menu selection over the other two interfaces.*

Table 8.31 summarizes the means, standard deviations and Shapiro-Wilks normality tests for the preference for Menu Selection.

Menu Selection		Midway			End		
Negative	Positive	M-Est	StDev	S-W	M-Est	StDev	S-W
Mouse	One	-3.8071	1.6432	.20	-1.5207	1.9083	.82
Mouse	Two	1.0756	3.6697	.30	1.2583	3.1937	.56
One	Two	1.8095	2.6646	.34	.6201	3.2711	.30

Table 8.31: All-subject double-ended comparisons of menu selection. A negative rating indicates preference for the interface in the column labeled *Negative*.

Table 8.32 shows tests for the null hypothesis for each of the menu selection ratings. The statistically significant result in this question is a clear preference for the Mouse-based menu selection mechanism over the One-Handed sundial menus, confirming half of hypothesis 28. This preference continues to the End questionnaire, with significance probability of 5.7%. However, since the preference moves towards neutrality at the End, there is an indication that the initial difficulties with using the menu are being trained out of existence. There is also a trend towards neutrality in the OneTwo case, indicating that which hand the menu is assigned to does not matter too much.

Coming on the heels of the previous question about menu appearance, this question might be misinterpreted as being the same question twice. In fact, the MouseTwo End comparison received the same answers for this question and the previous question.

Menu Selection	T-Test Significance	
Interfaces	Midway=0	End=0
MouseOne	.003	.057
MouseTwo	.675	.448
OneTwo	.226	.723

Table 8.32: Null hypothesis T-Tests for menu selection preference.

8.7 Detailed Comparison Discussion

While rating means change between Midway and End, in no question does the sign of a preference mean change from Midway to End. Table 8.33 presents a summary ranking of the three interfaces for the various interface components. This table has rearranged the order of the questions to group like sets of preferences together, with statistically significant results before nonsignificant results.

This ranking is generated by adding up the two End mean preferences that each interface will participate in, and ranking these three aggregate scores. The interface with highest score is placed in the *Most* column, with the next lower scores in the *Middle* and *Least* columns. The two exceptions to this are the Sundial menu questions. The Menu Selection question yields a statistically significant

Task	Most	Middle	Least
Menu Selection	**Mouse**	Two-Handed	One-Handed
Activating Rulers	**Mouse**	Two-Handed	One-Handed
Vertex Selection	Mouse	Two-Handed	One-Handed
Moving with Constraints	**Two-Handed**	Mouse	One-Handed
View Control	**Two-Handed**	One-Handed	Mouse
Examining Surface	**Two-Handed**	One-Handed	Mouse
Examining Rulers	Two-Handed	One-Handed	Mouse
Menu Appearance	No obvious preference overall		

Table 8.33: Summary of preferences for the various interface components. Boldface indicates a statistically significant result.

preference for the Mouse over the One-Handed system, and a nonsignificant preference of the Two-Handed system over the Mouse. Also, the OneTwo preference is quite small and nonsignificant, so the Mouse is ranked first. The Menu Appearance question does not yield a definite preference either for or against Sundial menus, hence the note in the last line of the table.

In most questions, the Mouse-based interface is at an extreme. It is viewed as being better than the other two interfaces, or worse. The exceptions to this are in Moving with Constraints, where the One-Handed interface places last. This is no doubt because of the high manual dexterity required in doing the constrained manipulation.

8.7.1 Discussion of Mouse Preferences

Subjects prefer Mouse-based interface for doing various selection tasks, namely vertex and menu selection. Activating rulers uses both vertex and menu selection. In other words, subjects preferred the mouse interaction techniques for operations that require picking from a relatively small set of choices. Subjects were probably disturbed by the noise in the tracker, and were more pleased by the apparently higher accuracy and precision of the Mouse.

Another common element to the One- and Two-Handed selection techniques is their use of orientation as a prime means of interaction. Compared to the other techniques, fine orientation control is necessary to achieve successful results in vertex and menu picking. Constrained reshaping in the One-Handed interface requires finer orientation control than the Two-Handed interface, because the right hand must be capable of orienting the bat and pressing the button in the One-Handed interface. In the Two-Handed interface, the left hand only needs to coarsely orient the left bat.

In the various examination tasks, the One- and Two-Handed interfaces are grouped favorably together. The orientation control in this case does not have a precision requirement.

Probably the most serious problem to fix in the 6 DOF tracker-based interfaces is Menu selection. Detailed preferences concerning it have the highest ratings in favor of the Mouse-based system, and my observations of the subjects during the trials indicates that some people find the Sundial difficult to use. Part of this can be overcome by the training regime outlined in section 5.7.2, but this training is not always going to be available, and some subjects had trouble even with training.

A lesser problem is vertex selection in 3D. People preferred the mouse-based system, and this

is probably due to the apparent directness of the interaction. The mouse cursor is always directly overhead the desired vertex, and so there is a strong visual coupling between the cursor and object that gives people the feeling of direct manipulation.

The probe does not offer this directness. Its very intent is to act upon a diffuse region of space and to pick the closest object, but there is not usually a strong occlusion cue indicating what the picked object should be. People would often confuse the probe axis with the snapping arrow, and would incorrectly assume that they were controlling the snapping arrow. A certain trust must be placed in the probe that it will do the right thing if it's pointed at the desired object.

8.7.2 Discussion of Two-Handed Preferences

One clear result of the detailed survey is that the Two-Handed system is always preferred on average to the One-Handed system. In particular, the one interaction technique that is explicitly two-handed yields the strongest preference over the One-Handed system.

Two-handed constrained reshaping allows the opportunity for a user to momentarily control 9 degrees of freedom at once. To do this, both the left hand and the right hand must be in motion – the left hand must be getting ready for the next constraint axis while the right hand is doing the current reshape operation. Once subjects gain confidence in the Two-Handed interface, I have observed a few of them doing just that.

Chapter 9

Experimental Results – Task Times

This chapter presents the task times that were completed by the subjects during the trials. Time trials are probably the least subjective measure of the interface, but this does not imply that these measurements are free of confounding aspects. One of the biggest problems with timed subject trials is the issue of errors.

Because I asked the subjects to go as fast as possible but make no errors, the time it took to correct any errors that were made along the way is included in the final time for a particular trial. When the error is easy to correct, this is not a problem. However, when the error is difficult to correct, the trial time ends up being much longer than it would otherwise be.

Unfortunately, one large error was rather easy to make in the One- and Two-Handed interfaces, which is when the nose and tail buttons of the right bat were pressed simultaneously. The nose button would activate vertex selection and the tail button would activate reshaping with orientation, usually resulting in a twisted surface. This error is tedious to correct, and since it comes as a surprise, the subject usually needs to be coached about what to do. This type of error caused some subjects to become flustered, and the resulting drop in confidence probably affected later trails.

For the first 14 subjects I thought that this potential problem should be left in the interface, since simultaneous button presses on the mouse would have a similar unexpected effect. I did not want to sanitize the bat-based interfaces, since the whole point of the user testing was that real tools with real functionality were being compared. Removing this problem seemed like cheating. I would therefore warn the subject to be careful in handling the right bat, and would explain how this error could occur. The problem with this approach was that subjects would make this error every now and then, and they would get caught in the tedious diversion of error correction, resulting in a useless timing value and a drop in confidence.

Moreover, the problem isn't really with the interface per se, but with the placement of the buttons on the bat. The error occurs because when the nose button is pressed by the index finger, the thumb is providing an opposing force against the finger's pressing force. If the thumb is on the tail button, the tail button also gets pressed. If the bat was redesigned to make this more difficult, then the error would be harder to make. Also, the software erroneously allowed simultaneous hits

to be interpreted as two independent nonoverlapping events. Disallowing chords would have fixed the problem without changing interface functionality, but I didn't think of doing this at the time. Instead, after the 14th subject, I simply disabled the reshaping with orientation operator. Given the tasks, the subject never notices this, and is left to make the less disastrous errors that arise from common carelessness.

Therefore, the first result of this section is that the bat buttons should be arranged to avoid having an opposing grip unintentionally cause two buttons to be pressed simultaneously. This is no doubt a standard piece of ergonomics wisdom.

9.1 Trial Results for 3x3 Grid

For task 1, 5 trials on Interface A were followed by 5 trials on Interface B. There were three pairs of interfaces, chosen from the three possible, and in each condition the presentation order was balanced. Therefore, a given interface is preceded or followed by the other two for half of the subjects. For the first block of trials, this is not likely to matter, because subjects only have exposure to interface A. For the second block, there might be some carryover effect from Interface A.

Mouse	MouseOne (3)		MouseTwo (3)		P Diff	Mouse (6)		
Trial	Mean	StDv	Mean	StDv	M-W Sig	M-Est	StDv	S-W
1	208.6	55.2	249.0	203.0	.51	170.7	134.9	.05
2	131.8	45.7	140.8	49.7	.51	109.0	43.0	.02
3	139.1	62.4	119.5	40.8	.51	99.6	48.3	.08
4	97.5	32.6	97.7	3.2	.51	90.7	20.7	.29
5	89.3	22.8	89.7	15.3	.83	88.4	17.4	.35
Min 1-5	89.3	22.8	86.1	11.8	.83	85.0	16.3	.35
6	137.1	53.9	142.3	48.4	.83	130.6	45.9	.08
7	123.7	40.3	89.9	27.7	.13	105.1	36.0	.12
8	98.6	16.7	94.1	27.9	.83	98.1	20.7	.71
9	85.7	7.0	78.9	15.4	.51	82.3	11.3	.40
10	73.4	9.0	85.3	19.1	.51	73.7	14.8	.24
Min 6-10	73.4	9.0	68.3	2.8	.27	68.5	6.6	.11

Table 9.1: Trial times in seconds for 3x3 grid – Mouse Interface. First two columns are for MouseOne condition, next two are MouseTwo condition, and P Diff column is the probability that these means are different. Last three columns are lumped **M-Estimates** for all Mouse trials. The Min lines contain the average of the minimum scores for the subject over the preceding 5 trials.

Table 9.1 shows the results for Task 1 for the Mouse-based interface. The two conditions are MouseOne and MouseTwo, with three subjects in each condition using the Mouse. The table shows the means and standard deviations for these two sets of 3 subjects, and shows the probability that these two sets have different means. Means are reported instead of M-Estimates because the estimator process would sometimes not converge. The test is a Mann-Whitney U test, which in this case will yield a result below .10 if all the times for one condition are less than or greater than the times for the other condition. The final 3 columns are the lumped Tukey biweight Mean Estimates for the Mouse-based interface, along with the overall standard deviation and Shapiro-Wilks normality test

result. The minimum time for each subject over a block of 5 trials is summarized in the Min lines. The horizontal line divides the first block (Interface A = Mouse) from the second block (Interface B = Mouse).

Table 9.1 shows that in block A and block B there is no difference in means between the two conditions of Mouse-based use. In other words, there is no difference in exposure effect between preceding use of the One- or Two-Handed systems. As expected, trial times improve as trials progress, and these numbers exhibit the power law of practice [13]. The falling trend in standard deviations also indicates that the subjects are closing in on a "true" mean task time after a certain number of trials.

One	MouseOne (3)		OneTwo (3)		P Diff	One (6)		
Trial	Mean	StDv	Mean	StDv	M-W Sig	M-Est	StDv	S-W
1	216.1	90.2	111.8	30.5	.25	149.3	87.0	.06
2	197.4	30.8	95.2	13.9	.08	157.0	60.5	.27
3	138.3	56.0	80.9	1.8	.56	78.8	50.5	.02
4	100.0	29.1	81.4	13.3	.56	91.8	23.9	.36
5	105.9	31.9	66.5	6.9	.25	67.6	31.4	.07
Min 1-5	100.0	29.1	66.5	6.9	.25	67.5	27.8	.10
6	165.0	73.2	78.7	23.2	.04			
7	118.9	41.6	82.5	5.9	.04			
8	105.3	46.7	70.5	14.8	.28	73.9	36.4	.04
9	89.5	11.0	67.3	6.9	.04			
10	106.5	20.4	58.4	12.2	.08			
Min 6-10	83.6	10.4	60.9	9.6	.08			

Table 9.2: Trial times in seconds for 3x3 grid – One-Handed Interface. First two columns are for MouseOne condition, next two are OneTwo condition, and P Diff column is the probability that these means are different. The Min lines contain the average of the minimum scores for the subject over the preceding 5 trials. Last three columns are lumped M-Estimates for all One-Handed trials. The second block of trials seems to have two different means dependent on condition.

Table 9.2 shows the results for Task 1 for the One-Handed interface. The two conditions are MouseOne and OneTwo, with three subjects in each condition using the One-Handed interface. The table shows the means and standard deviations for these two sets of 3 subjects, and shows the Mann-Whitney U test probability that these two sets have different means. The final 3 columns are the lumped M-Estimates for the One-Handed interface, along with the overall standard deviation and Shapiro-Wilks normality test result.

The lumped means columns are blank for the second block because there are evidently two different means for the One-Handed interface depending on condition. The MouseOne condition exhibits similar behavior as was shown in table 9.1, where the times rapidly fall in trials 1 through 5. Trial 6 yields the same value as trial 2, then trials 7 through 10 execute a similar descent. However, trials 6 through 10 of the OneTwo condition continue the downward progress of trials 1 through 5 as though the same interface were in use by the subjects in both blocks of trials. The Mann-Whitney U tests for difference in component means across trials 6, 7, 9, and 10 all yield significance probabilities that indicate that a significant difference between the means.

Two	MouseTwo (3)		OneTwo (3)		P Diff	Two (6)		
Trial	Mean	StDv	Mean	StDv	M-W Sig	M-Est	StDv	S-W
1	157.2	46.5	143.6	33.1	.83	141.4	36.8	.42
2	105.7	12.4	113.8	17.5	.83	107.5	14.3	.26
3	118.9	33.8	95.3	15.4	.28	103.2	26.8	.62
4	89.7	32.7	101.4	11.5	.51	95.2	22.8	.74
5	98.0	36.0	87.1	13.3	.83	85.7	25.0	.42
Min 1-5	75.0	16.2	80.9	10.6	.51	77.1	12.6	.23
6	170.9	83.8	102.5	62.9	.28	114.9	76.1	.78
7	136.1	35.4	101.3	70.4	.51	118.4	53.4	.29
8	114.5	17.1	74.1	40.3	.28	101.0	35.4	.48
9	82.0	5.3	123.3	126.2	.51	69.9	83.0	.01
10	117.7	20.1	81.8	51.5	.51	102.2	40.1	.19
Min 6-10	82.0	5.3	69.7	42.4	.51	75.0	27.9	.78
	MouseTwo (3)		OneTwo (2)		P Diff			
6	170.9	83.8	69.8	38.5	.08			
7	136.1	35.4	60.6	1.7	.08			
8	114.5	17.1	52.1	18.2	.08			
9	82.0	5.3	50.5	1.8	.08			
10	117.7	20.1	52.0	2.3	.08			
Min 6-10	82.0	5.3	45.5	8.9	.08			

Table 9.3: Trial times in seconds for 3x3 grid – Two-Handed Interface. First two columns are for MouseTwo condition, next two are OneTwo condition, and P Diff column is the probability that these means are different. Last three columns are lumped M-Estimates for all Two-Handed trials. The Min lines hold the average of the minimum scores for the subject over the preceding 5 trials. The middle block is trials 6-10 with all subjects. The last block excludes one subject from the OneTwo condition.

Since this is the OneTwo condition, clearly the cause of these lower than expected trial times is prior exposure to the Two-Handed interface. In using the One-Handed interface, subjects are able to take substantial parts of their immediately-preceding Two-Handed experience and apply it to the One-Handed interface with only minor modification. The consistently small standard deviations for each condition is further evidence that a training effect has occurred.

Table 9.3 shows the results for Task 1 for the Two-Handed interface. The two conditions are MouseTwo and OneTwo, with three subjects in each condition using the Two-Handed interface. The table shows the means and standard deviations for these two sets of 3 subjects, and shows the Mann-Whitney U test probability that these two sets have different means. The final 3 columns are the lumped M-Estimates for the Two-Handed interface, along with the overall standard deviation and Shapiro-Wilks normality test result.

The component means and the Mann-Whitney significances for Interface B seem to be in conflict with each other. The OneTwo mean is somewhat less than the MouseTwo mean most of the time, but the OneTwo condition has very large standard deviations. In fact, trial 9's standard deviation is larger than the mean, indicating that a negative trial time is statistically plausible! The reason for this is that one of the subjects in the OneTwo condition had a difficult time with both the One-Handed and the Two-Handed interfaces. This subject's minimum time for all of the trials on Task 1 was 118 seconds, which is longer than all but the first trial mean using the two-handed interface. If this subject is excluded, then the means again exhibit a training effect, as shown in the bottom block of table 9.3. That is, the prior use of the One-Handed interface serves to train the user on the Two-Handed interface, and the means for the Two-Handed interface in the second block of trials are seemingly a continuation of the first block of trials, as if the interfaces haven't changed.

This is not against expectations, but it makes the analysis of the second block of trials more cumbersome, because it can be interpreted as an example asymmetric transfer that Poulton warns against [50, 51]. Poulton's method of dealing with this issue is to conduct a between-subjects experiment. That is, have subjects use only one interface, which in the case of this experiment means discarding the results of the second block of trials.

The issue is not quite as clear-cut as in the case that Poulton is considering. In that case, two treatments A and B are being compared, and A's influence on B is differently than B's influence on A. In this case, three interfaces are involved, and are grouped in pairs AB, AC and BC. Within each pair there seems to be symmetric influence, in that there is no difference between AB and BA. However, between pairs this is not the case, because the One-Handed and Two-Handed interfaces influence each other more strongly than the Mouse-based system. Since this differential influence occurs only in the OneTwo condition, the OneTwo results should be isolated from the other results. The trial times for the OneTwo condition can only be compared against each other. Similarly, the MouseOne and MouseTwo conditions can only be compared against each other, and not against the OneTwo times.

9.1.1 First Block Task 1 Performance

In the first block of trials, there is no differential training issue to worry about, so times for the same interface under different conditions can be grouped together. Table 9.4 shows the lumped M-Estimates for the first 5 trials for each interface, and the probabilities for differences between the means. The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Task1A	M-Estimates			Tests		
Trial	Mouse	One	Two	Mouse-One	Mouse-Two	One-Two
1	170.7	149.3	141.4	.465	.262	1.000
2	109.0	157.0	107.5	.715	.262	.160T
3	99.6	78.8	103.2	.273	.423	.855
4	90.7	91.8	95.2	.717T	.875T	.836T
5	88.4	67.6	85.7	.584	.748	.583
Min 1-5	85.0	67.5	77.1	.937T	.274T	.509T

Table 9.4: Tests for differences between means for first block of grid trials. A letter T after a significance probability indicates a T-test was used, otherwise a Mann-Whitney U test was used.

The test results indicate that there is not a significant difference between the trial times for each interface. The lowest probability is associated with the One-Two comparison in trial 2, but in this case, the One-Handed interface had an anomalously large mean which was larger than the preceding trial 1 mean, indicating that a few subjects spent extra time fixing errors during this trial. Overall, subjects performed equally well on all three interfaces, and improved their scores at about the same rate.

The real significance of this result lies in the fact that the 3x3 grid task is a wholly 2D task, for which the tools and operations available in the Mouse-based interface have been optimized. The One-Handed and Two-Handed interfaces are not really intended for strictly 2D manipulations, but subjects can perform them successfully with these tools after only a small amount of training.

Recalling section 6.2.8, only 16 minutes extra training and practice time is required to introduce the non-Mouse interfaces. The results here indicate that subjects can perform as well in this interface after only 16 extra minutes.

9.1.2 MouseOne and MouseTwo Task 1 Performance

Table 9.5 shows the lumped M-estimates for the second set of 5 trials for the MouseOne and MouseTwo conditions. In this table, the One-Handed and Two-Handed times are Tukey Biweight Mean estimators, instead of the means that were used in tables 9.2 and 9.3.

Except for trial 10 and the minimum over the last 5 trials, the results of these tests are the same as in section 9.1.1. There is no significant difference between the interfaces for trials 6 through 9. Trial 10 shows a significant difference between the Mouse-based interface and the One- and Two-Handed interfaces, but in both cases the non-mouse mean is greater than the previous trial mean, indicating that some subjects were taking extra time to fix errors during these trials.

The minimum score does not have this problem, of course, so the significant differences between

Task1B	M-Estimates			Tests		
Trial	Mouse	One	Two	Mouse-One	Mouse-Two	One-Two
6	130.6	133.9	144.6	.439	.439	.827
7	105.1	101.7	134.2	.439	.302	.275
8	98.1	78.5	108.3	.796	.302	.513
9	82.3	89.5	85.0	.439	.796	.275
10	73.7	95.9	117.7	.071	.039	.275
Min 6-10	68.5	83.6	82.0	.056T	.039T	.824T

Table 9.5: Tests for differences between means for second block of grid trials excluding the OneTwo case.

the Mouse and the non-Mouse interfaces have some importance. The major difference between the second block and the first block of trials is that the subject is quite familiar with the task. All subjects achieved a minimum trial time on or after trial 8, so the subject has seen it performed twice, has practiced it twice, and has performed it at least 8 times in timed trials. The subject is also somewhat more familiar with the mouse-based interface than with the One- and Two-Handed interfaces, as the Midway questionnaire that followed this block of trials confirmed.

One way to look at this is to examine the increase in task time from the minimum of trials 1 through 5 to trial 6, when subjects switch interfaces. Table 9.6 shows the means of these differences. Except in the OneTwo condition, where there is a strong training effect, the mean overall difference is 66.1 seconds. The mean difference for the OneTwo condition is 6.5 seconds. This difference in times from the first block to trial 6 is due to the new interface being comparatively unfamiliar. By the end of the second block of trials, the effect of this difference in familiarity is at most 15 seconds.

Trial 6-5	Mouse		One		Two		All	
Trial Order	M1	M2	1M	12	2M	21	Non-12	12
Difference	75.7	84.7	37.1	15.9	67.3	-2.2	66.1	6.5

Table 9.6: Mean difference between minimum trial 1-5 time and trial 6 time.

Compared to the distribution of minimum scores for the first block of trials, it is evident that the improved Mouse score in the second block is due to the Mouse-based interface being more familiar to the subject.

In fact, if we examine the minimum scores for the three interfaces for both the first and second block of trials, excluding the second block of the OneTwo condition, we get a total of 30 scores, as shown in table 9.7. This considers the times for both interfaces for each subject. That is, the MouseOne and MouseTwo subjects contribute two minimum scores, and the OneTwo subjects contribute one minimum score.

The Mouse score is 5.3 seconds less than Two-Handed score, and 6.3 seconds less than the One-Handed score, but these differences are not statistically significant.

Task 1	M-Estimates			Tests		
Interface	Mouse	One	Two	Mouse-One	Mouse-Two	One-Two
Times	73.9	80.2	79.2	.616	.722	.482T

Table 9.7: Comparisons of minima from both blocks of task 1 excluding the second block of the OneTwo case.

9.1.3 OneTwo Condition Task 1 Performance

Table 9.8 shows the One-Handed and Two-Handed times for the second block of the OneTwo case. Trials 2 and 4 yield tests with low probabilities, indicating that the Two-Handed subjects might be completing the task more quickly than the One-Handed subjects. However, the test of the minimum scores indicates that there is not a significant difference between the means.

Task1B	M-Estimates		Tests
Trial	One	Two	One-Two
6	78.7	69.8	.564
7	82.5	60.6	.083
8	70.5	52.1	.248
9	67.1	50.5	.083
10	64.1	52.0	.564
Min 6-10	60.9	45.5	.248

Table 9.8: Tests for differences between means for second block of grid trials for the OneTwo case.

Again the result is that given the same stage of practice, there is no difference between the One-Handed and the Two-Handed interfaces in terms of trial completion time.

If we ignore the stage of practice, there is one last set of comparisons to make for task 1, and that is to compare the times for the second block of the OneTwo case to the other minima. Table 9.9 shows these tests. The 3 columns headed *OneTwo Block B* contain the minima for the One-Handed, Two-Handed, and both interfaces respectively in the OneTwo case. Each line in the table is for one of the interfaces not from the second block of the OneTwo condition. The Mouse A+B line is all of the mouse times for the first and second blocks of task 1. The Mouse B line is the minima for the second block of trials, so this comparison is the most appropriate, because the amount of task practice is equal between the One- and Two-Handed case and the Mouse. The lines labeled One and Two are the minima for the MouseOne and MouseTwo conditions, respectively.

The result is that there is a statistically significant difference between the OneTwo block B minimum times and all of the other minimum times except for the comparison of MouseB against One-Handed. The implication is that if the subjects are given enough practice in the interface, they can complete the task quite quickly.

I would argue for a stronger interpretation of this result by recalling the fact that the subjects have an average of 1.9 years prior mouse experience and at least 3 months of AutoCAD experience, and that this is a 2D task. By their own report, Subjects feel more familiar with the Mouse-based interface. The fact that in the OneTwo case they can perform this task faster by a statistically

Task 1		OneTwo Block B			Non-OneTwo versus:		
Non-OneTwo Interface		One	Two	Both	One	Two	Both
MouseA+B	73.9	60.9	45.5	54.8	.043	.029	.006
MouseB	68.5	60.9	45.5	54.8	.197	.046	.033T
One	80.2	60.9	45.5	54.8	.041	.037	.015T
Two	79.2	60.9	45.5	54.8	.042	.033	.002T

Table 9.9: Comparisons of minima from the second block of the OneTwo condition against the lumped minima for all other conditions. MouseA+B is all mouse times, while MouseB is just the minima from block B.

significant margin indicates that the non-mouse interfaces are competitive or superior to the Mouse-based interface.

9.2 Trial Results for Folding a Box

For task 2, 5 trials on Interface B were followed by 5 trials on Interface A. For most subjects, this program was followed as intended, but there were some problems with completion for some subjects. The major problem was that I had scheduled the appointments for the subjects too closely together in the first two days, so one subject's box folding trials would spill over into the next subject's appointed time. A small spillover was acceptable, because the new subject can spend the first few minutes filling in the first four pages of the survey. However, once this overlap was used up, the new subject needed to start training, which meant that the current subject needed to finish. The luck of the draw would have it that a few of these early subjects were in the OneTwo case, so the missing trial times are concentrated in this condition.

Table 9.10 shows the results for Task 2 for the Mouse-based interface. The two conditions are MouseOne and MouseTwo, with three subjects in each condition using the Mouse. The table shows the means and standard deviations for these two sets of 3 subjects, and shows the probability that these two sets have different means. Means are reported instead of M-Estimates because the estimator process would sometimes not converge. The test is a Mann-Whitney U test, which in this case will yield a result below .10 if all the times for one condition are less than or greater than the times for the other condition. The final 3 columns are the lumped Tukey biweight Mean Estimates for the Mouse-based interface, along with the overall standard deviation and Shapiro-Wilks normality test result. Because some subjects did not complete all trials, the Num column indicates the number of subjects that are included in the lumped M-estimate. Some subjects were missing for the first block of Mouse trials, but not the second block. The minimum time for each subject over a block of 5 trials is summarized in the Min lines. The horizontal line divides the first block (Interface B = Mouse) from the second block (Interface A = Mouse).

Table 9.10 shows that in the first and second blocks there is no difference in means between the two conditions of Mouse-based use. In other words, there is no difference in exposure effect between preceding use of the One- or Two-Handed systems. Part of the reason for large standard deviations for the MouseOne condition is that one of the subjects had a difficult time with the second task using both assigned interfaces. This is not the same subject who had difficulty in the first task.

Mouse	MouseOne		MouseTwo		P Diff	Num	Mouse		
Trial	Mean	StDv	Mean	StDv	Sig.	Subj	M-Est	StDv	S-W
1	256.1	52.2	153.9	40.1	.13	6	203.8	69.7	.38
2	171.3	19.8	103.6	9.0	.05	6	136.0	39.5	.37
3	129.9	15.7	92.0	14.6	.08	5	106.6	24.5	.39
4	124.7	31.0	112.9	64.1	.56	5	109.6	48.3	.26
5	118.0	31.7	138.8	90.2	.99	4	119.9	56.5	
Min 1-5	129.5	52.5	86.3	18.1	.28	6	91.7	42.4	.05
6	270.5	75.0	167.7	18.0	.05	6	172.3	74.5	.04
7	207.0	87.3	142.9	20.8	.28	6	148.5	66.7	.07
8	221.1	67.1	110.9	5.7	.05	6	118.7	73.9	.05
9	155.6	88.4	113.3	19.9	.82	6	109.1	61.8	.01
10	143.3	87.5	94.7	13.6	.51	6	94.2	62.0	.01
Min 6-10	138.5	90.4	92.8	10.3	.83	6	90.5	62.7	.01

Table 9.10: Trial times in seconds for Box folding – Mouse Interface. First two columns are for the MouseOne condition, next two are the MouseTwo condition, and P Diff column is the probability that these means are different. Last three columns are lumped M-Estimates for all Mouse trials. The Num column is the number of subjects contributing to the lumped mean for that line. The Min lines contain the average of the minimum scores for the subject over the preceding 5 trials.

In fact, this first subject managed to overcome the difficulties and perform reasonably well in the second task.

As evidenced by the large standard deviations and by the lack of a clear decreasing trend in the component means, the Box Folding task is somewhat more difficult than the 3x3 grid task. The mean minimum completion time for the Mouse-based interface is 90 seconds, which is 25 seconds longer than the 3x3 grid task.

Table 9.11 shows the results for Task 2 for the One-Handed interface. The two conditions are MouseOne and OneTwo, with nominally three subjects in each condition using the One-Handed interface. The final 3 columns are the lumped Tukey biweight Mean Estimates for the One-handed interface, along with the overall standard deviation and Shapiro-Wilks normality test result. The Num column indicates how many subjects were included in the lumped estimate. The second block of trials was particularly hard hit, with only two subjects completing the last trial.

The Mann-Whitney U tests for differences between the component means do not indicate a difference due to previous exposure to one of the interfaces. This confirms the findings of section 6.2.8 that by the time the last set of trials is to be started, the user is familiar enough with the each interface that immediately preceding exposure has no effect.

Table 9.12 shows the results for Task 2 for the Two-Handed interface. The two conditions are MouseTwo and OneTwo, with nominally three subjects in each condition using the Two-Handed interface. The final 3 columns are the lumped Tukey biweight Mean Estimates for the Two-handed interface, along with the overall standard deviation and Shapiro-Wilks normality test result. The num column indicates the number of participating subjects, with some gaps in the first block, but complete participation in the second block.

The Mann-Whitney U tests for differences between the component means do not indicate a

One	MouseOne		OneTwo		P Diff	Num	One		
Trial	Mean	StDv	Mean	StDv	Sig.	Subj	M-Est	StDv	S-W
1	279.3	168.0	226.3	133.1	.28	6	165.9	138.6	.04
2	315.5	50.0	217.1	142.2	.51	6	275.6	109.5	.33
3	260.3	118.0	204.6	75.2	.51	6	220.9	93.6	.48
4	197.1	105.7	156.5	96.8	.56	5	161.6	91.8	.43
5	162.1	79.5	203.6	124.2	.82	6	167.2	96.0	.42
Min 1-5	162.1	79.5	140.9	50.1	.82	6	141.9	60.5	.53
6	497.1	104.8	218.8	44.1	.08	5	384.0	170.9	.50
7	219.3	41.3	234.5	50.4	.82	6	226.7	42.1	.27
8	230.8	82.6	226.8	92.2	.99	4	229.0	71.5	
9	192.7	45.4	285.0		.22	3	223.6	62.2	
10	145.4		158.0		.32	2	151.7	8.9	
Min 6-10	169.7	29.9	166.0	10.9	.83	6	160.6	20.2	.41

Table 9.11: Trial times in seconds for Box folding – One-Handed Interface. First two columns are for the MouseOne condition, next two are the OneTwo condition, and P Diff column is the probability that these means are different. Last three columns are lumped M-Estimates for all One-Handed trials. The Num column is the number of subjects contributing to the lumped mean for that line. The Min lines contain the average of the minimum scores for the subject over the preceding 5 trials.

Two	MouseTwo		OneTwo		P Diff	Num	Two		
Trial	Mean	StDv	Mean	StDv	Sig.	Subj	M-Est	StDv	S-W
1	207.7	31.5	283.9	92.3	.28	6	217.7	74.5	.30
2	166.5	16.6	259.3	45.4	.08	5	187.0	56.9	.26
3	173.7	9.8	180.8		.65	4	175.1	8.8	
4	167.5	19.6	202.5	72.8	.52	6	166.5	51.4	.16
5	135.9	25.5	198.0		.18	4	149.3	37.4	
Min 1-5	135.9	25.5	172.0	31.3	.13	6	153.0	32.3	.72
6	231.9	109.8	208.1	99.1	.83	6	203.7	94.5	.50
7	170.3	68.4	240.0	171.0	.83	6	137.1	122.6	.02
8	195.6	122.3	173.3	115.2	.52	6	116.0	106.9	.04
9	114.6	15.9	174.1	78.2	.52	6	116.9	60.1	.04
10	103.6	13.7	126.5	52.2	.83	6	100.3	36.4	.04
Min 6-10	103.6	13.7	126.5	52.2	.83	6	100.3	36.4	.04

Table 9.12: Trial times in seconds for Box folding – Two-Handed Interface. First two columns are for the MouseTwo condition, next two are the OneTwo condition, and P Diff column is the probability that these means are different. Last three columns are lumped M-Estimates for all Two-Handed trials. The Num column is the number of subjects contributing to the lumped mean for that line. The Min lines contain the average of the minimum scores for the subject over the preceding 5 trials.

difference due to previous exposure to one of the interfaces. This confirms the findings of section 6.2.8 that by the time the last set of trials is to be started, the user is familiar enough with the each interface that immediately preceding exposure has no effect.

Task 2	M-Estimates			Tests		
Trial	Mouse	One	Two	Mouse-One	Mouse-Two	One-Two
1	203.8	165.9	217.7	.749	.351T	.423
2	136.0	275.6	187.0	.034T	.049T	.280T
3	106.6	220.9	175.1	.021T	.014	.394
4	109.6	161.6	166.5	.210T	.053T	.927T
5	119.9	167.2	149.3	.393	.564	.999
Min 1-5	91.7	141.9	153.0	.150	.037	.933T
6	172.3	384.0	203.7	.045	.749	.071T
7	148.5	226.7	137.1	.078	.810	.149
8	118.7	229.0	116.0	.088	.999	.394
9	109.1	223.6	116.9	.071	.688	.121
10	94.2	151.7	100.3	.182	.631	.182
Min 6-10	90.5	160.6	100.3	.055	.423	.037

Table 9.13: Tests for differences between means for the box trials. A letter T after a significance probability indicates a T-test was used, otherwise a Mann-Whitney U test was used.

Unlike task 1, there is no need to cut the analysis up into two parts, because the training effect apparent in task 1 does not exist here. Table 9.13 shows the lumped M-Estimates for the first and second block of 5 trials with their minimum times for each interface, and the probabilities for differences between the means. The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

The test results indicate that subjects can complete the box folding task fastest on the Mouse-based system. Compared to the Two-Handed interface, 3 of the 5 trials and the trial minima have a statistically significant difference for the first block. Compared to the One-Handed interface, 2 of the 5 trials have a statistically significant difference from the Mouse-based system, and the minimum time has a low probability for the first block. However, the probability that the One-Handed and Two-Handed means are the same is quite high, indicating that the best interpretation is that the Mouse-based system is faster than both non-Mouse systems for the first block of trials.

The first interesting point in the second block is that the mouse-based minimum is the same as for the first block. The One-Handed times are longer, and the Two-Handed times are somewhat shorter than for the previous block of trials. The test results indicate that the subjects can complete the box folding task about equally quickly on the Two-Handed and the Mouse-based system. None of the significance values comparing these interfaces falls below .423, indicating that this trend is quite solid.

By contrast, the One-Handed scores are somewhat slower than the other two, as indicated by low probability values in the Mouse-One and One-Two columns.

In the Mouse-One column, the first trial has a statistically significant difference, and the minimum time is quite close to the significance level of 0.05. All but one of the remaining trials have probabilities less than 0.1. In the One-Two column, the first trial is low, and the minimum times

yield a statistically significant result.

If the minimum scores for the three interfaces for both the first and second block of trials are lumped together, we get a total of 36 scores, as shown in table 9.14. This considers the times for both interfaces for each subject. That is, each subject contributes two scores to this table.

Task 2	M-Estimates			Tests		
Interface	Mouse	One	Two	Mouse-One	Mouse-Two	One-Two
Times	91.3	155.3	131.3	.010	.057	.150T

Table 9.14: Comparisons of minima from both blocks of task 2.

The comparison of the Mouse time to the One-Handed time is strongly significant, indicating that the Mouse-based time is quite definitely shorter than the One-Handed time. The Two-Handed comparison seems to be next. The comparison to the Mouse-Based interface is not quite significant, and the comparison to the One-Handed interface has a low probability, indication that perhaps the One-Handed interface is the slowest of all.

The reason for this is that Task 2 is extremely easy to do using the Mouse-based interface. It consists of a number of orthogonal 2D operations that are performed in each of the 3 orthographic windows. For example, the desired edge to fold is selected in the top window, which has a plan view of the 3x3 plus shape. Next, the selected edge is moved up 1 meter in the front window, and moved to the left 1 meter in the side window. This first edge requires care to make sure that it is the correctly located, while the following edges can rely on the correctness of this edge. The 2 edges to the left and right are done next. Each edge is selected in the top window, and rotated up to match the standing edge in the front window, since the front window sees these 2 edges edge-on. The last edge is selected in the front window, and rotated up in the side window. As one subject said in the post-test interview, performing the Mouse-based test was like performing a task by rote, or as if the subject was executing a program. The point being that the strong performance in the Mouse-based interface is due as much to this factor as to any innate advantage offered by the interface.

9.3 Summary

In summary, the performance of the two tasks yielded two rather surprising results. In task 1, a wholly 2D task of creating a 3x3 grid of rectangles and stretching them out to be square, the best times were performed on the Two-Handed interface followed by the One-Handed interface. This is surprising, because the subjects had two years of Mouse experience, and 3 months of AutoCAD experience to draw upon. They had no 6 DOF tracker experience, and only 16 minutes extra training and practice time to achieve this result.

In task 2, subjects using the Mouse-based interface turned in the best performance, followed by the Two-Handed interface and the One-Handed interface. This is a 3D task, but because it consists of a collection of axially-aligned manipulations, it can be done more easily with the 3-view drawing of the mouse-based system than with the free-space operations provided by the Two-Handed and One-Handed interfaces.

Chapter 10

Summary and Conclusions

This document has presented the design and implementation of a two-handed 3D design system called THRED that allows for rapid sketching and refinement of polygonal free-form surfaces. Its output subsystem is a high-performance graphics workstation that renders the scene in realtime. Its input subsystem is the pair of 6 DOF bats. Neither of these subsystems encumber the user, and the effect of the output subsystem on user comfort is quite well characterized [57]. One of the contributions of this work is to identify if THRED induces pain and fatigue in the user, and the results of Chapter 7 indicate that it does not.

The work presented in this document can be broken into two parts – the interaction techniques needed to make the interface work, and the evaluation of the interface.

10.1 Interface Evaluation

THRED was constructed with the idea that the underlying surface editor could have more than one interface. This allows the comparison of interfaces without confounding factors, because they are based on the same underlying operations.

With this study, I have built three interfaces to the underlying database and operations available in THRED. The interfaces are THRED, a Mouse-Based interface similar to the Alias modeler [1], and a One-Handed interface similar to JDCAD [41, 42].

I conducted a user test of the three interfaces to evaluate their effectiveness and ease of use. A total of 18 students who had completed a first course in AutoCAD learned two of the three editing styles in practice sessions, and performed a total of 20 trials – 10 on each interface. There were thus three experimental conditions:

1. Mouse-Based and One-Handed (MouseOne).
2. Mouse-Based and Two-Handed (MouseTwo).
3. One-Handed and Two-Handed (OneTwo).

Subjects were instructed to create a simple regular 3x3 grid from a single square, and then to fold a box from this grid. The simplicity of the object made the goal easy for the subjects to understand

and attain. Subjects also filled in a questionnaire outlining their experience, their preferences, and their level of pain and fatigue.

This is an unbiased framework for the evaluation of user interfaces. The order of presentation of interfaces is counterbalanced, and there was an equal number of subjects randomly assigned to each of the three interface styles. The three interface styles are interfaces to the same underlying editor, with the same operations available in each interface. The subjects perform an identical task on each interface, and the step-by-step operations needed to perform the task are the same in each interface. The difference between the interfaces is in the input devices, and in the detailed physical manipulations of these devices required to generate a syntactically valid operation.

In addition to task invariance, other elements of this experimental framework are the characterization of the subjects' experience, the measurement of the training and practice times, and the evaluation of pain and fatigue. These survey questions are presented in a counterbalanced manner where appropriate, and are repeated so as to reduce the chances of a subject's extreme reactions skewing the results.

Taken as a whole, this is a new experimental framework for interface evaluation that extends the current state of user testing. Currently, most user tests seek to characterize a small aspect of the interface [35, 74, 78]. The goal of this experimental framework is to characterize the user interface in the large, and it succeeds in the goal because of its comprehensive coverage of evaluation methods.

10.1.1 Times

The subjects' trial times indicate that on a strictly 2D task, THRED users can outperform users of the competing systems. Considering the much higher level of familiarity that subjects had with Mouse-Based systems, this is a strong result. It means that the despite the slowness of the 6 DOF trackers (100 milliseconds lag), and the prototypical nature of the buttons placed on them, users are able to outperform a mature well-established, well-understood technology. It also illustrates that the complexities implicit in the 3D Two-Handed style can be readily understood by the user.

For the box-folding task, the Mouse-Based system and the Two-Handed system performed about equally well, while the One-Handed system was somewhat slower. This is because this task can be conveniently broken up into a series of convenient 2D orthogonal operations which exactly coincide with the 3 orthographic views of the Mouse-Based interface. The first Mouse-Based subject discovered this, and I was obliged to tell all following Mouse-Based subjects about this technique to avoid biasing the experimental results.

As one subject commented later, performing the box-folding task with the Mouse-Based interface was simply a rote procedure. With the Two-Handed and One-Handed interfaces, subjects still perform a real 3D task, so these results indicate that the only way that the Mouse-Based interface is competitive with THRED in performing 3D tasks is if a similar rote procedure is developed.

10.1.2 Learning

The subjects had at least 3 months of formal training in AutoCAD, and an average of 1.9 years of exposure to the mouse. No subject had any 6 DOF tracker experience whatsoever. This gives

the Mouse-Based interface significantly greater level of familiarity than the Two-Handed interface, as noted in two ways. First, training and practice times for the Mouse-Based interface were shorter than for the non-Mouse systems. Second, subjects reported a significantly higher familiarity rating for the Mouse-Based interface than for the non-Mouse interfaces at the Midway survey.

However, this 3 months of extra training and 1.9 years extra experience offered slight advantage. For the non-mouse interfaces, only 16 more minutes of training and practice time was required. For training, the extra time was used to introduce the new interface style, the new tracking technology and the new interface techniques never before seen by the subject. The extra practice time was used by the subjects to get basic operational familiarity with this brand-new technology. Immediately after this, in the first 3x3 grid trial, Mouse-Based system subjects were able to accomplish the task in 171 seconds, One-Handed system subjects took 149 seconds, and Two-Handed system subjects took 141 seconds. Large variances prevent statistically significant difference in the means. However, the central message is that the advantages of 3 months of AutoCAD and 1.9 years of mouse usage have been overcome by a mere 16 minutes of extra training and practice.

One can expect that as users gain more exposure to 6 DOF tracker-based interfaces, the necessity for extra training will disappear, and users will be immediately able to outperform equivalent Mouse-Based interfaces using 6 DOF trackers.

10.1.3 Pain and Fatigue

This is the first survey of pain or fatigue effects related to the use of 6 DOF tracking systems outside the study of simulation sickness. The importance of this is simply that one of the commonplace objections to the extended use of 6 DOF trackers is that arm pain or fatigue might set in. Because users interact with bat-based systems with their hands not resting on the table, and because some muscular effort is required to maintain the hand's presence above the table, this is a sensible concern. One of my main motivators for conducting the pain and fatigue survey was to allay these concerns, to quantify what areas of the body might be most at risk of discomfort, and to compare interface styles with respect to their discomfort level.

The results of this survey indicate that THRED presented minimal difficulties in the realm of user pain and fatigue, while the One-Handed interface induced statistically significant fatigue. In order of increasing pain and fatigue, the subjects in the MouseTwo condition experienced the least, followed by MouseOne subjects. The OneTwo subjects had the most pain and fatigue, with statistically significant fatigue increases over the course of the experiment. This leads me to conclude that the Two-Handed interface has as small an impact on user pain and fatigue as the Mouse-Based interface. The importance of this result is simply that commonplace fears about the use of 6 DOF trackers for an extended period of time are unfounded if the Two-Handed interface style is used. The Two-Handed interface style is as painless as the Mouse-Based style.

In the general design space of 6 DOF user interfaces, the Two-Handed style successfully offers a comfortable user interface that can be quickly learned. Based on the success of the Two-Handed style, and on the ability of users to quickly accomplish the given tasks, I offer these design guidelines for future 3D user interfaces:

1. The user should sit in a chair in front of a standard workstation console.

2. The user should sit in a chair that has arms, so that he or she may rest his or her elbows on the arms of the chair while holding the trackers aloft.

3. The interface hardware should be a pair of small lightweight 6 DOF position and orientation trackers with buttons, with one tracker assigned to each hand. The left hand should be assigned low-frequency context-setting tasks, and the right hand should perform picking and fine manipulation.

4. Interaction techniques must not require that the user maintain a static hand or arm posture.

5. The interface should provide position and orientation recentering commands that the user can quickly invoke. This allows the user to place the bats at the most comfortable location, and issue recentering commands to customize the interface to suit the user's needs.

6. If a user becomes uncomfortable with the bats at a particular location, the user should change locations and again recenter the bats.

The first two recommendations are about the ergonomics of the workplace. The next recommendation proposes the user interface style, and a concise explanation of the way in which tasks are assigned to each tracker. Recommendations 4 and 5 offer guidance about how to construct the interaction system so as to avoid user fatigue. Recommendation 6 is for the user.

10.1.4 Preferences

In general, subjects preferred THRED the most, and the One-Handed interface the least. Nowhere in the preference survey is there evidence to suggest that subjects preferred the One-Handed interface over THRED.

The preference survey indicated that there is room for improvement in THRED, particularly in the area of menu selection. The trial times also indicate that the 3-view drawing technique used by most CAD systems offers some capabilities that are worth investigating in THRED.

These preferences bode well for the future of the Two-Handed interface style. Subjects felt that for these tasks, the Two-Handed interface provided a more comfortable, more pleasing style than the One-Handed or Mouse-Based styles. Subjects completed the preference survey twice, so during the second survey, they had complete knowledge of the survey questions, and could make ratings in light of this knowledge. In view of the detailed ratings which criticized some elements of the Two-Handed system, it is clear that it is the integration of the interaction elements that make the Two-Handed interface appealing.

10.2 Interaction Style

The interaction style of THRED is that the user holds a bat in each hand, and uses these bats for all input. The left hand provides context, and the right hand performs picking and fine manipulation.

To put some meat on these bones, I developed some interaction techniques that generate context in various senses of the word and allow picking and fine manipulation to be performed.

The basic interaction technique in THRED is the manipulation of the position and orientation of the entire scene. This provides the spatial context for all other operations, and is assigned exclusively to the left hand because it provides spatial context. The basic techniques for the right hand are picking vertices and reshaping the surface.

The picking operation utilizes a new technique called the Probe, which uses a specialized metric to evaluate all candidates, selecting the candidate with the minimum value.

Unconstrained reshaping moves selected leaf-level vertices in a way that induces one of a small set of shapes on the surface. When parent vertices are selected, vertices at a finer level of refinement ride on the surface of the coarse-level quadrilaterals. This allows broad-scale editing while preserving finer details in a way that is different from Forsey's system [25].

Constrained reshaping uses a new bimanual technique that employs the left hand to constrain the motion of the right hand's manipulation of the surface. The left hand chooses the axis or plane that the right hand will move along, and the right hand selects the distance to move. The technique is natural for inexperienced users to pick up, and in user tests, some subjects were able to overlap left and right hand motion. This technique allows the user to state 9 degrees of freedom in one operation.

For setting modes and invoking commands, a new menu technique called the sundial menu is used. When the user presses a bat button, this circular menu pops up with the selectable items arrayed in sectors about the center. At the center is the selection cursor called the shadow stick, which the user pivots by reorienting the bat to select the desired item. The menu allows users to select graphical objects and text items, and is hierarchical, potentially allowing thousands of menu choices.

One of the menu choices in THRED invokes a new visualization technique called textured rulers which allow the user to visually measure the lengths of polygon edges. These rulers can be applied to any polygon, and if the user desires, the entire surface can be textured in this way, allowing hundreds of measurements to be available simultaneously.

10.3 The Thesis

The thesis of this work is that for 3D computer-aided design of polygon-based free-form surfaces, two hands are better than one.

Evaluating the Mouse-Based interface, we have the following:

- THRED users outperform users of the Mouse-Based interface on a simple but not trivial 2D task.
- THRED users get equal performance to users of the Mouse-Based interface on a simple but not trivial 3D task.
- General subject preference is for THRED.

- With detailed preferences, the Mouse-Based interface is preferred for menu selection and vertex picking. Otherwise, THRED was preferred.
- Pain and fatigue results show that THRED induces equal short-term discomfort to the Mouse-Based interface.
- Training times are at most 16 minutes more than the Mouse-Based interface, ignoring parallel learning effects.

The equal performance of THRED and the Mouse-Based interface in the 3D task can be explained by noting that it can be broken into a rote-learned series of 2D operations ideally set up for the 3 orthographic views of the Mouse-Based interface. Moreover, subjects *did not need to perform window navigation* after the first trial in the Mouse-Based interface. Subjects simply used the window configuration of the previous trial, and so could optimize out any window setup time. Had I insisted that the subject start from the same startup position, there arises the issue of what is a fair startup position. To avoid this issue, subjects could continue from the previous trial.

Therefore, I argue that the short task times for the Mouse-Based interface are best-case times, because no thinking is necessary, and the scene can be set up to maximize speed. With THRED, no such ideal setup is possible, and the subjects must learn true 3D manipulation.

I therefore claim that THRED is better than the Mouse-Based interface for certain types of 2D tasks, and is at least equal to the Mouse-Based interface for simple 3D tasks.

To generalize this statement, one must examine what was compared. The Mouse-Based interface was selected from among the best professional CAD systems commercially available. The Mouse-Based interface is a clone of the Alias modeler, which has been carefully developed to maximize user productivity for a real complex problem domain. Alias's interaction has been carefully designed to minimize extraneous mouse hits and maximize throughput. The intent of this experimental comparison was to compare the new technology against the toughest level of competition.

Therefore, I claim that THRED is equal or better to the Mouse-Based CAD interfaces as long as THRED can maintain real-time interaction. For models that are too complex to draw in real time, THRED ceases being truly functional. For hierarchical B-Spline surfaces, Forsey speeds up drawing by displaying a coarse polygonal approximation. These and other techniques would serve to widen the applicability of the THRED interface style.

The central claim of this work is that THRED is as effective as mouse-based systems of similar complexity for 3D tasks, and is more effective than mouse-based systems for 2D tasks. This claim is supported by the experimental results.

For two-handed interfaces, the experimental results support the idea that the hands should be given complementary roles. Although no other two-handed style was compared, the equal or superior performance of THRED compared to the other two systems at least indicates that this organization schema is not a poor one. That is to say, there is no evidence against this organization schema.

Evaluating the One-Handed interface, we have the following:

- THRED users get equal performance to users of the One-Handed interface on a simple but not trivial 2D task.
- THRED users outperform users of the One-Handed interface on a simple but not trivial 3D task.
- All general and detailed preferences are for THRED.
- Pain and fatigue results show that THRED induces less discomfort than the One-Handed interface by a statistically significant margin.
- Training times are about equal for both interfaces.

The conclusions here are somewhat more clear-cut. THRED outperforms the One-Handed interface on all but the 2D task.

To generalize, new One-Handed interfaces must certainly avoid the ergonomic problems involved with static keyboard postures. However, I cannot make the same argument that this One-Handed interface is the best one existing, so it is difficult to say if THRED would outperform any One-Handed interface. I will argue that for a similar set of functionality, the two 6 DOF trackers provide extra degrees of freedom on which to hang some of the spatial functionality of the interface, thereby making it easier to use. The One-Handed interface simply moved all spatial functions to the right bat, and users found this more difficult to use.

10.4 Future Work

The results of the preference survey show the way to areas of future work. Menu selection is still a partially solved problem in 3D user interfaces. More detailed design and testing in this area is warranted.

The selection of surface features using the probe is effective in areas of low depth complexity, or with sparse populations of objects. However, with many densely-packed small objects, it is likely that this technique will break down due to high depth complexity along the probe axis, making the desired object difficult to pick. This more complex domain warrants study.

The bat presented in section 4.2 is an improvement over commercial systems, but is still not ergonomically ideal. Accidental button presses due to opposing pressure on two buttons caused problems in the One- and Two-Handed trials, so a better design is needed.

With respect to the experimental evaluation, there are five general areas of improvement. First, there was no control condition in this experiment that could be used as a comparative basis for some of the measurements. For example, there is a statistical test called Moses Extreme Reactions [63], which tests if survey responses in an experimental condition are more extreme than in the control condition. This would help explain if some of the survey variances are reasonable. A sensible control would be to use two different Mouse-Based editors, one based on Alias, and one based on another commercial tool. This would also help to better characterize the trial-to-trial learning by establishing a learning baseline.

Second, the overall experimental time needs to be somewhat longer. Training and practice take a significant amount of time, and this needs to be accounted for. Also, a longer experiment would allow more trials to be completed, allowing users to close in on their asypmtotically best time. The trial time variances get smaller trial by trial, and this process would probably continue until standard deviations reached 5-10 seconds or so.

Third, a better task than 3D box folding should be used. The problem with this task is that it devolves to a series of rote manipulations in the Mouse-Based interface. A better task would call for the creation of a more geometrically complex object that would be simple to describe. This object should not be easily created using many identical axis-aligned 2D steps. Verification would again be done by examining the measurement textures and grids.

Fourth, a better way of handling errors is needed. The major difficulty with doing large tasks like box folding is that some errors will seriously impede task completion. Lengthy error correction saps the user's confidence, magnifying the effect of the error. An experimental protocol that fairly scores the effect of errors while allowing the user to confidently continue should be developed. This would narrow the variance in the One-Handed and Two-Handed interfaces enough to give a statistically significant result. Of course the other obvious approach worth taking is to make errors harder to make by improving the interfaces.

Finally, the other significant generally-mentioned roadblock to general acceptance of this technology is training. The general assumption that people would make is that THRED is hard to learn. The results of this work indicate that this is not true, but it would be good to get a tighter bound on training and practice. The present experimental system lumps interface training and task training together, since that seemed like the most efficient way to do it (as verified by the timing analysis). However, perhaps task training can be somehow separated from interface training in a two step process that allows the user to first learn a extremely simple dummy task. The dummy task would allow the user to get used to the feel of THRED. Next, the real task would be introduced, and the user would be able to quickly apply the newfound interface knowledge.

10.5 Conclusions

Increasing spatial input bandwidth cannot be done without regard to the ergonomics involved. The pain and fatigue surveys clearly indicate that THRED and the Mouse-Based interface yield low discomfort, which contradicts the established wisdom that bat-based interfaces are likely to be painful or fatiguing to use. My work in evaluating THRED indicates that THRED is as ergonomically sound as Mouse-Based interfaces.

The timing results of my experiment indicate that THRED is competitive with the state of the art in 2D interfaces in a 2D task after only 16 extra minutes of training and practice.

In terms of performance, the importance of this is twofold. First, the 6 DOF tracking technology that THRED used in these experiments is 10 years old and somewhat slow. Lag and noise in sensor technology usually has a deleterious effect on user performance, so better performance with poorer tracking technology indicates that 6 DOF tracker-based performance can only improve. Second, the task consisted of operations entirely in the X-Y plane. Over 30 years of research has gone into

the technologies of 2D interfaces, and the Mouse-Based system is in some sense the culmination of that work. The fact that subjects with no experience in 6 DOF trackers can outperform the Mouse-Based system on its home ground indicates that the performance gap between THRED and the Mouse-Based systems will only get wider on more strongly 3D tasks.

In terms of learning, the 16 extra minutes of extra training and practice required for THRED was more than sufficient to overcome the experience advantage of the Mouse-Based system. All subjects had 3 month of formal training on AutoCAD, and an average of 1.9 years experience in using a mouse. Yet, after 16 minutes more training, these subjects were able to accomplish the first trial of the first 2D task 30 seconds faster with THRED than with the Mouse-Based interface.

The future of 3D user interfaces lies in the thoughtful integration of interaction techniques while bearing in mind the potential drawbacks of the technology. The work I have presented here shows that systems that follow the THRED style will maximize the positive impact while holding the negative impact to a minimum.

Bibliography

[1] Alias Research Inc. *Alias/3 User's Manual*. Toronto, 1993.

[2] Alan H. Barr. Global and Local Deformations of Solid Primitives. In Hank Christiansen, editor, *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, pages 21–30, July 1984.

[3] Dominique Bechmann. Space Deformation Models Survey. *Computers and Graphics*, 18(4):571–586, 1994.

[4] Eric A. Bier, Maureen C. Stone, Ken Fishkin, William Buxton, and Thomas Baudel. A Taxonomy of See-Through Tools. In Beth Adelson, Susan Dumais, and Judith Olson, editors, *Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems*, pages 358–364, 1994.

[5] Eric Allan Bier. Skitters and Jacks: Interactive 3D Positioning Tools. In Frank Crow and Stephen M Pizer, editors, *Proceedings of ACM 1986 Workshop on Interactive 3D Graphics*, pages 183–196, Chapel Hill, North Carolina, 1986. ACM SIGGRAPH.

[6] Alison Black. Visible Planning on Paper and on Screen: The Impact of Working Medium on Decision-Making by Novice Graphic Designers. *Behaviour and Information Technology*, 9(4):283–296, 1990.

[7] P. Borrel and D. Bechmann. Deformation of n-dimensional objects. *Internat. J. Comput. Geom. Appl.*, 1(4):427–453, 1991.

[8] Paul Borrel and Ari Rappoport. Simple Constrained Deformations for Geometric Modeling and Interactive Design. *ACM Transactions on Graphics*, 13(2):137–155, April 1994.

[9] M.L. Braunstein. *Depth Perception Through Motion*. Academic Press, New York, 1976.

[10] J. Butterworth, A. Davidson, S. Hench, and T.M. Olano. 3DM: A Three Dimensional Modeler Using a Head-Mounted Display. In *Proceedings of 1992 Symposium on Interactive 3D Graphics*, pages 135–138, Cambridge, Massachusetts, March 29 - April 1 1992. ACM SIGGRAPH.

[11] William Buxton and Brad A. Myers. A Study in Two-Handed Input. In *Proceedings of ACM CHI'86 Conference on Human Factors in Computing Systems,*, pages 321–326, Boston, Massachusetts, April 13-17 1986.

[12] Jack Callahan, Don Hopkins, Mark Weiser, and Ben Shneiderman. An Empirical Comparison of Pie vs. Linear Menus. In *Proceedings of ACM CHI'88 Conference on Human Factors in Computing Systems*, pages 95–100. ACM SIGCHI, Washington, DC, 1988.

[13] Stuart K. Card, Thomas P. Moran, and Allen Newell. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1983.

[14] Michael Chen, S. Joy Mountford, and Abigail Sellen. A Study in Interactive 3-D Rotation Using 2-D Control Devices. *Computer Graphics (Proceedings of SIGGRAPH '88)*, 22(4):121–129, August 1-5 1988.

[15] James H. Clark. Designing Surfaces in 3-D. *Communications of the ACM*, 19(8):454–460, August 1976.

[16] Sabine Coquillart. Extended Free-Form Deformation: A Sculpturing Tool for 3D Geometric Modeling. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 187–196, August 1990.

[17] Michael Deering. The HoloSketch VR Sketching System. *Communications of the ACM*, 39(5):54–63, May 1996.

[18] Michael Deering. High Resolution Virtual Reality. *Computer Graphics (Proceedings of SIGGRAPH '92)*, 26(2):195–202, July 26-31 1992.

[19] Kenneth B. Evans, Peter P. Tanner, and Marceli Wein. Tablet-based Valuators that Provide One, Two, or Three Degrees of Freedom. *Computer Graphics (Proceedings of SIGGRAPH '81)*, 15(3):91–97, August 1981.

[20] Gerald Farin. *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide*. Academic Press, New York, 1988.

[21] P. M. Fitts. The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. *Journal of Experimental Psychology*, 47:381–391, 1954.

[22] J. D. Foley, V. L. Wallace, and P. Chan. The Human Factors of Computer Graphics Interaction Techniques. *IEEE Computer Graphics and Applications*, 4(11):13–48, November 1984.

[23] James D Foley, Andries van Dam, Steven K Feiner, and John F Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, Reading, Mass., 1990.

[24] James. D. Foley and Victor L. Wallace. The Art of Natural Graphic Man-Machine Conversation. In *Proceedings IEEE 62, 4*, pages 462–471, April 1974.

[25] David Forsey and Richard Bartels. Hierarchical B-Spline Refinement. *Computer Graphics (Proceedings of SIGGRAPH '88)*, 22(4):205–212, August 1-5 1988.

[26] Barry Fowler. Geometric manipulation of tensor product surfaces. In David Zeltzer, editor, *Computer Graphics (1992 Symposium on Interactive 3D Graphics)*, volume 25(2), pages 101–108, March 1992.

[27] Josef Griessmair and Werner Purgathofer. Deformation of Solids with Trivariate B-Splines. In W. Hansmann, F. R. A. Hopgood, and W. Strasser, editors, *Eurographics '89*, pages 137–148. North-Holland, September 1989.

[28] Yves Guiard. Asymmetric Division of Labor in Human Skilled Bimanual Action: The Kinematic Chain as a Model. *The Journal of Motor Behavior*, 19(4):486–517, 1987.

[29] Kenneth P Herndon, Robert C Zeleznik, Daniel C Robbins, D. Brookshire Conner, Scott S Snibbe, and Andries van Dam. Interactive Shadows. In *UIST 1992 Proceedings*, pages 1–6, 1992.

[30] Ken Hinckley, Randy Pausch, John C. Goble, and Neal F. Kassel. Passive Real-World Interface Props for Neurosurgical Visualization. In Beth Adelson, Susan Dumais, and Judith Olson, editors, *Human Factors in Computing Systems CHI'94 Conference Proceedings*, pages 452–458, Boston, Massachusetts, April 24-28 1994. ACM SIGCHI.

[31] William M. Hsu, John F. Hughes, and Henry Kaufman. Direct manipulation of free-form deformations. In Edwin E. Catmull, editor, *Computer Graphics (SIGGRAPH '92 Proceedings)*, volume 26, pages 177–184, July 1992.

[32] Hiroo Iwata. Artificial Reality with Force-feedback: Development of Desktop Virtual Space with Compact Master Manipulator. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 165–170, August 1990.

[33] T. W. Jensen, C. S. Petersen, and M. A. Watkins. Practical curves and surfaces for a geometric modeler. *Computer Aided Geometric Design*, 8(5):357–370, 1991.

[34] K. I. Joy. Utilizing parametric hyperpatch methods for modeling and display of free-form solids. *Internat. J. Comput. Geom. Appl.*, 1(4):455–471, 1991.

[35] Paul Kabbash and William Buxton. The "Prince" Technique: Fitts' Law and Selection Using Area Cursors. In Mary Beth Rosson and Jakob Nielsen, editors, *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 1, pages 273–279, Denver, Colorado, May 7-11 1995.

[36] Paul Kabbash, William Buxton, and Abigail J Sellen. Two-Handed Input in a Compound Task. In Beth Adelson, Susan Dumais, and Judith Olson, editors, *Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems*, volume 1, pages 417–423, 1994.

[37] Paul Kabbash, I. Scott MacKenzie, and William Buxton. Human Performance Using Computer Input Devices in the Preferred and Non-Preferred Hands. In Stacey Ashlund, Kevin Mullet, Austin Henderson, Erik Hollnagel, and Ted White, editors, *Human Factors in Computing Systems INTERCHI'93 Conference Proceedings*, pages 474–481, Amsterdam, April 24-29 1993. ACM SIGCHI.

[38] J.W. Kling, Lorrin A. Riggs, and 17 contributors. *Woodworth and Schlosberg's Experimental Psychology*. Holt, Rinehart and Winston, Inc., New York, 3rd edition edition, 1971.

[39] Gordon Kurtenbach and William Buxton. User Learning and Performance with Marking Menus. In Beth Adelson, Susan Dumais, and Judith Olson, editors, *Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems*, volume 1, pages 258–264, 1994.

[40] Gordon P Kurtenbach, Abigail J Sellen, and William A S Buxton. An Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus. *Human-Computer Interaction*, 8(1):1–23, 1993.

[41] Jiandong Liang and Mark Green. JDCAD: A Highly Interactive 3D Modeling System. *Computers and Graphics*, 18(4):499–506, 1994.

[42] Jiandong Liang and Mark Green. Geometric Modeling Using Six Degrees of Freedom Input Devices. In *3rd International Conference on CAD and Computer Graphics*, pages 217–222, Beijing, China, August 23-26, 1993.

[43] Logitech Inc. *2D/6D Mouse Technical Reference Manual*. Fremont, California, 1991.

[44] Mark Mitchell and Janina Jolley. *Research Design Explained*. Harcourt Brace Jovanovich College Publishers, Fort Worth, Texas, 1992.

[45] Gregory M. Nielson. CAGD's top ten: What to watch. *IEEE Computer Graphics and Applications*, 13(1):35–37, January 1993.

[46] Gregory M. Nielson and Dan R. Olsen. Direct Manipulation Techniques for Objects Using 2D Locator Devices. In Frank Crow and Stephen M Pizer, editors, *Proceedings of 1986 ACM Workshop on Interactive 3D Graphics*, pages 175–182, Chapel Hill, North Carolina, October 1986,. ACM SIGGRAPH.

[47] Marija J. Norusis. *SPSS for UNIX : base system user's guide, release 5.0*. SPSS Inc., Chicago, Ill., release 5.0 edition, 1993.

[48] R C Oldfield. The Assessment and Analysis of Handedness: The Edinburgh Inventory. *Neuropsychologia*, 9:97–113, 1971.

[49] Timothy Poston and Luis Serra. The Virtual Workbench: Dextrous VR. In Gurminder Singh, Steven K Feiner, and Daniel Thalmann, editors, *Virtual Reality Software and Technology (VRST 94)*, pages 111–121, Singapore, August 23-26 1994. World Scientific.

[50] E C Poulton. *Tracking Skill and Manual Control.* Academic Press, New York, 1974.

[51] E C Poulton. Influential Companions. Effects of one strategy on another in the within-subjects designs of cognitive psychology. *Psychological Bulletin*, 91:673–690, 1982.

[52] M Rettig. Prototyping for Tiny Fingers. *Communications of the ACM*, 37(4):21–27, April 1994.

[53] Emanuel Sachs, Andrew Roberts, and David Stoops. 3-Draw: A Tool for Designing 3D Shapes. *IEEE Computer Graphics and Applications*, 11(6):18–24, November 1991.

[54] H. Samet. The Quadtree and Related Hierarchical Data Structures. *ACM Comp. Surv.*, 16:187–260, 1984.

[55] Chris Schmandt. Spatial Input/Output Correspondence in a Stereoscopic Computer Graphics Work Station. *Computer Graphics (Proceedings of SIGGRAPH '83)*, 17(3):253–261, July 1983.

[56] Thomas W. Sederberg and Scott R. Parry. Free-Form Deformation of Solid Geometric Models. In David C. Evans and Russell J. Athay, editors, *Computer Graphics (SIGGRAPH '86 Proceedings)*, volume 20, pages 151–160, August 1986.

[57] Don Sellers. *ZAP! How your computer can hurt you – and what you can do about it.* Peachpit Press, Berkeley, California, 1994.

[58] Luis Serra, Timothy Poston, Hern Ng, Beng Choon Chua, and John A. Waterworth. Interaction Techniques for a Virtual Workspace. In *Virtual Reality Software and Technology (VRST/ICAT 95)*, Chiba, JAPAN, 1995.

[59] John J Shaughnessy and Eugene B Zechmeister. *Research Methods in Psychology.* McGraw-Hill Publishing Company, New York, 1990.

[60] Chris Shaw and Mark Green. Two-Handed Polygonal Surface Design. In *UIST 1994 Proceedings*, pages 205–212, Marina del Rey, California, November 2-4 1994. ACM SIGGRAPH/SIGCHI.

[61] Chris Shaw, Mark Green, Jiandong Liang, and Yunqi Sun. Decoupled Simulation in Virtual Reality with The MR Toolkit. *ACM Transactions on Information Systems*, 11(3):287–317, July 1993.

[62] Ken Shoemake. ARCBALL: A User Interface for Specifying Three-Dimensional Orientation Using a Mouse. In *Proceedings of Graphics Interface '92*, pages 151–156, May 1992.

[63] Sidney Siegel and John Castellan Jr. *Nonparametric statistics for the behavioral sciences.* McGraw-Hill, New York, 2nd edition edition, 1988.

[64] E Sittas. 3D Design Reference Framework. *Computer-Aided Design*, 23(5):380–384, June 1991.

[65] Scott S Snibbe, Kenneth P Herndon, Daniel C Robbins, D. Brookshire Conner, and Andries van Dam. Using Deformations to Explore 3D Widget Design. *Computer Graphics (Proceedings of SIGGRAPH '92)*, 26(2):351–352, July 26-31 1992.

[66] Richard Stoakley, Matthew J. Conway, and Randy Pausch. Virtual Reality on a WIM: Interactive Worlds in Miniature. In Mary Beth Rosson and Jakob Nielsen, editors, *Proceedings of ACM CHI'95 Conference on Human Factors in Computing Systems*, volume 1, pages 265–272, Denver, Colorado, May 7-11 1995.

[67] Demetri Terzopoulos. Regularization of Inverse Visual Problems Involving Discontinuities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(4):413–424, 1986.

[68] Demetri Terzopoulos and Kurt Fleischer. Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture. In John Dill, editor, *Computer Graphics (SIGGRAPH '88 Proceedings)*, volume 22, pages 269–278, August 1988.

[69] Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. Elastically Deformable Models. In Maureen C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21, pages 205–214, July 1987.

[70] Demetri Terzopoulos and Andrew Witkin. Physically-Based Models with Rigid and Deformable Components. In *Proceedings of Graphics Interface '88*, pages 146–154, June 1988.

[71] Demetri Terzopoulos and Andrew Witkin. Physically Based Models with Rigid and Deformable Components. *IEEE Computer Graphics and Applications*, 8(6):41–51, November 1988.

[72] Maarten J. G. M. van Emmerik. Interactive Design of 3D Models with Geometric Constraints. *Computer Graphics Forum*, 7(5-6):309–325, September 1991.

[73] Colin Ware, Kevin Arthur, and Kellogg S. Booth. Fishtank Virtual Reality. In Stacey Ashlund, Kevin Mullet, Austin Henderson, Erik Hollnagel, and Ted White, editors, *Human Factors in Computing Systems INTERCHI'93 Conference Proceedings*, pages 37–42, Amsterdam, April 24-29 1993. ACM SIGCHI.

[74] Colin Ware and Glenn Franck. Evaluating Stereo and Motion Cues for Visualizing Information Nets in Three Dimensions. *ACM Transactions on Graphics*, 15(2):121–140, April 1996.

[75] Colin Ware and Danny R. Jessome. Using the Bat: A Six Dimensional Mouse for Object Placement. In *Proceedings of Graphics Interface '88*, pages 119–124, Edmonton, Alberta, June 6-10, 1988.

[76] Y. Y. Wong. Layer Tool: Support for Progressive Design. In Stacey Ashlund, Kevin Mullet, Austin Henderson, Erik Hollnagel, and Ted White, editors, *Human Factors in Computing Systems INTERCHI'93 Adjunct Proceedings*, pages 127–128, Amsterdam, April 24-29 1993. ACM SIGCHI.

[77] Shumin Zhai, William Buxton, and Paul Milgram. The "Silk Cursor": Investigating Transparency for 3D Target Acquisition. In Beth Adelson, Susan Dumais, and Judith Olson, editors, *Proceedings of ACM CHI'94 Conference on Human Factors in Computing Systems*, volume 1, pages 459–464. ACM SIGCHI, 1994.

[78] Shumin Zhai, Paul Milgram, and William Buxton. The Influence of Muscle Groups on Performance of Multiple Degree-of-Freedom Input. In *Proceedings of ACM CHI'96 Conference on Human Factors in Computing Systems*, pages 308–315. ACM SIGCHI, April 13-18 1996.

[79] Thomas G. Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. A Hand Gesture Interface Device. In *CHI + GI 1987, Human Factors in Computing Systems and Graphics Interface*, pages 189–192, April 5-9 1987.

Appendix A

Pain and Fatigue Tables

This appendix lists tests for the null hypothesis with respect to the pain and fatigue survey. The results from this appendix are summarized in chapter 7, and identical section titles and numbers are used so that the correspondence is easy to find. The tables that appear in chapter 7 are duplicated here for completeness.

A.1 Head and Torso Fatigue

Fatigue All Region	Midway=0 2-tail P	End=0 2-tail P
Eyes/Face	.4413	.3863
Neck	.4008	.3863
Shoulders	.4990	.2340
Back	.6121	.6726

Table A.1: Null hypothesis tests for all-subject head and torso fatigue scores.

Table A.1 shows that the probabilities for the Midway=0 hypothesis are all greater than 0.05, indicating that the Midway scores are not likely different from 0. The same result exists for the End scores, indicating that they are also not likely different from 0. The acceptance of the null hypothesis for midway and end scores indicates that these two sets of scores do not differ from each other.

Table A.2 shows the scores for the Mouse and One-Handed interface condition, ignoring presentation order.

Fatigue MouseOne	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	.2011	1.6330	.53	-.5854	2.9668	.14
Neck	.0391	.5431	.56	.2037	.4082	.01
Shoulders	.4259	1.2111	.01	.5741	1.6130	.01
Back	.2943	1.0464	.14	-.0358	.9683	.37

Table A.2: Fatigue statistics for head and torso for the MouseOne condition.

Because some of these data appear to be normally distributed, the T-Test can be used to test

the null hypothesis. The blanks in the T-Test columns indicate that the data did not appear to be normal, and so the Wilcoxon test was used instead. Testing the null hypothesis for these means is shown in table A.3.

Fatigue MouseOne	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Eyes/Face	.638	.706		
Neck	.529			.3863
Shoulders			.4990	.2340
Back	.911	.506		

Table A.3: Null hypothesis tests for head and torso fatigue in the MouseOne condition.

Again all of the test results are greater than 0.05, indicating that the null hypothesis should be accepted.

Table A.4 shows the scores for the MouseTwo condition.

Fatigue MouseTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	-.2015	.7528	.25	.1296	.9832	.06
Neck	0.0000	.6325	.20	.0833	1.0400	.53
Shoulders	-.0421	.8165	.02	-.0631	.8093	.04
Back	-.2593	.8165	.01	.0000	.6325	.20

Table A.4: Fatigue statistics for head and torso for the MouseTwo condition.

Testing the null hypothesis for these means is shown in table A.5.

Fatigue MouseTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Eyes/Face	.611			.6547
Neck	1.000	.489		
Shoulders			.1088	.1088
Back		1.000	.3173	

Table A.5: Null hypothesis tests for head and torso fatigue in the MouseTwo condition.

Again all of the test results are greater than 0.05, indicating that the null hypothesis should be accepted.

Table A.6 shows the scores for the OneTwo condition. Testing the null hypothesis for these means is shown in table A.7. Again all of the test results are greater than 0.05, indicating that the null hypothesis should be accepted.

All of these previous tests have dealt with the issue of whether the null hypothesis can be contradicted, which would indicate a trend in the direction of increased or decreased fatigue. However, none of tests give evidence against the null hypothesis, so we can assume that hypotheses 10, 11, and 12 can be accepted.

Fatigue OneTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	-.1886	1.2247	.09	.5661	1.7889	.55
Neck	.0615	1.0328	.48	.0000	.8944	.13
Shoulders	.7037	1.6021	.01	1.1125	1.3292	.05
Back	.3779	1.1690	.04	.4444	.8367	.01

Table A.6: Fatigue statistics for head and torso for the OneTwo condition.

Fatigue OneTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Eyes/Face		.229	.2850	
Neck	.465	1.000		
Shoulders			.1797	.1088
Back			.1088	.1797

Table A.7: Null hypothesis tests for head and torso fatigue in the OneTwo condition.

In dealing with hypothesis 13, comparisons between means for each condition should be made. A total of three comparisons can be made, for each pair of conditions using a Mann-Whitney test or a T-Test if the data are normally distributed. The headings of table A.8 indicate condition pairings:

- M1-M2 = MouseOne - MouseTwo
- M1-12 = MouseOne - OneTwo
- M2-12 = MouseTwo - OneTwo

Table A.8 presents tests for difference of Midway fatigue scores, with means briefly summarized. The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Fatigue Midway	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Eyes/Face	.20	-.20	-.18	.511T	.932	.342
Neck	.03	0.00	.06	.669T	.708T	.515T
Shoulders	.42	-.04	.70	.032	.849	.032
Back	.29	-.25	.37	.340	.263	.045

Table A.8: Midway fatigue comparisons between conditions for head and torso.

For the Midway fatigue scores, the Eyes/Face and Neck scores are similar for all three conditions. The Shoulder scores for the MouseTwo condition are significantly less than for other two conditions. Comparing the OneTwo and MouseOne conditions yield a probability that indicates similar means for these two conditions.

The Back scores for the MouseTwo condition are significantly less than the OneTwo score only. The other tests indicate that the MouseOne scores could be the same as either or both of the other two scores.

Table A.9 presents tests for difference of End fatigue scores, with means briefly summarized. Again, the test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Fatigue End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Eyes/Face	-.58	.12	.56	.864	.722T	.391
Neck	.20	.08	.00	.668	.605	.584T
Shoulders	.57	-.06	1.11	.032	.591	.021
Back	-.03	.00	.44	.562T	.153	.293

Table A.9: Midway fatigue comparisons between conditions for head and torso.

For the End fatigue scores, the Eyes/Face, Neck and Back scores are similar for all three conditions.

The Shoulder scores for the MouseTwo condition are significantly less than for other two conditions. Comparing the OneTwo and MouseOne conditions yield a probability that indicates similar means for these two conditions.

In summary, where the statistical tests indicate a difference between conditions, ranking in order of increasing scores is as follows:

1. MouseTwo
2. MouseOne
3. OneTwo

Hypothesis 13 is false for the Shoulders in both surveys after the start of the experiment, and false for the Back during the Midway survey. This means that the interface seems to have an effect on subject head and torso fatigue. However, analyzing all fatigue ratings and fatigue by condition, none of the means exceed 0 by a statistically significant margin, so it seems that head and torso fatigue increase only slightly.

A.2 Head and Torso Pain

Table A.10 indicates head and torso pain ratings for all subjects, with Shapiro-Wilks normality probabilities.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.11.

All of the means seem quite small, and none show any significant difference from 0. Analyzing by condition yields table A.12 for the MouseOne case.

Testing for the null hypothesis yields table A.13.

Pain All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	.1840	.7471	.01	.1963	.9750	.01
Neck	.2407	.6401	.01	.0648	.6913	.01
Shoulders	.3519	1.0273	.01	.1728	1.1163	.01
Back	.1358	.7699	.01	.0846	.9450	.01

Table A.10: All-subject pain statistics for head and torso.

Pain All	Midway=0	End=0
Region	2-tail P	2-tail P
Eyes/Face	.4469	.3454
Neck	.2626	.5541
Shoulders	.1508	.3452
Back	.5754	.7794

Table A.11: Null hypothesis tests for all-subject head and torso pain scores.

Pain MouseOne	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	.9509	1.0926	.53	-.1335	1.5513	.32
Neck	.2963	.4916	.01	.3796	.6646	.01
Shoulders	.2461	1.1690	.03	.6296	1.6021	.01
Back	.4222	.6765	.11	-.0667	.8385	.32

Table A.12: Pain statistics for head and torso for the MouseOne condition.

Pain MouseOne	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Eyes/Face	.225	.363		
Neck			.1797	.1797
Shoulders			.1088	.1797
Back	.224	.353		

Table A.13: Testing null hypothesis for head and torso pain in the MouseOne condition.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.14 is for the MouseTwo case.

Pain MouseTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	-.1296	.4082	.01	.0000	.6325	.20
Neck	.2015	.7528	.25	-.2015	.7528	.25
Shoulders	-.1296	.9832	.06	-.2593	.8165	.01
Back	-.1296	.9832	.06	.3778	1.0954	.09

Table A.14: Pain statistics for head and torso for the MouseTwo condition. Caption

Testing for the null hypothesis yields the table A.15.

Pain MouseTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Eyes/Face		1.000	.3173	
Neck	.611	.611		
Shoulders			.6547	.3173
Back			.6547	1.0009

Table A.15: Testing null hypothesis for head and torso pain in the MouseTwo condition.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.16 is for the OneTwo case.

Pain OneTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Eyes/Face	.1296	.4082	.01	.1296	.4082	.01
Neck	.2015	.7528	.25	.0000	.6325	.20
Shoulders	.4444	.8367	.01	.3148	.5164	.01
Back	.0000	.6325	.20	-.1296	.9832	.06

Table A.16: Pain statistics for head and torso for the OneTwo condition.

Testing for the null hypothesis yields table A.17.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

All of these previous tests have dealt with the issue of whether the null hypothesis can be contradicted, which would indicate a trend in the direction of increased or decreased pain. However, none of tests give evidence against the null hypothesis, so we can assume that hypotheses 10, 11, and 12 can be accepted.

For hypothesis 13, pairwise comparisons are again made.

Table A.18 presents tests for difference of Midway pain scores, with means briefly summarized.

Pain OneTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Eyes/Face			.3173	.3173
Neck	.611	1.000		
Shoulders			.1797	.1797
Back	1.000			.6547

Table A.17: Testing null hypothesis for head and torso pain in the OneTwo condition.

The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Pain Midway	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Eyes/Face	.95	-.13	.13	.104	.263	.176
Neck	.30	.20	.20	.857	.857	1.000T
Shoulders	.25	-.13	.44	.180	.655	.296
Back	.42	-.13	.00	.389	.335T	.924

Table A.18: Midway pain comparisons between conditions for head and torso.

Since all of the significance probabilities are greater than 0.100, all of the Midway pain scores are similar for all three conditions.

Table A.19 presents tests for difference of End pain scores, with means briefly summarized. Again, test results ending in T indicate that a T-Test was used.

Pain End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Eyes/Face	-.13	.00	.13	.387T	.775	.598
Neck	.38	-.20	.00	.179	.293	.687T
Shoulders	.63	-.26	.31	.093	1.000	.092
Back	-.07	.38	-.13	.796	.531	.652

Table A.19: End pain comparisons between conditions for head and torso.

Since all of the significance probabilities are greater than 0.090, all of the Midway pain scores are similar for all three conditions.

In summary, there is no statistically significant difference between conditions for any of the pain ratings for the head and torso. Hypothesis 13 therefore holds for head and torso pain.

A.3 Left Arm Fatigue

Table A.20 indicates left arm fatigue ratings for all subjects, with Shapiro-Wilks normality probabilities.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.21.

In this case, the left hand means both seem to differ significantly from zero. As the following

Fatigue All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.7222	1.6118	.01	.7346	1.2984	.01
Left Wrist	.5494	1.7352	.01	.5247	1.3735	.01
Left Arm	.3364	1.7176	.01	.2191	1.4445	.01

Table A.20: All-subject fatigue statistics for the left arm.

Fatigue All	Midway=0	End=0
Region	2-tail P	2-tail P
Left Hand	.0382	.0218
Left Wrist	.0747	.0630
Left Arm	.4017	.6121

Table A.21: Null hypothesis tests for all-subject left arm fatigue scores.

analysis will show, this is partly because of a difference in means between conditions. The standard deviations for these scores are larger than for the head and torso fatigue, which also indicates a difference between conditions.

Analyzing by condition yields table A.22 for the MouseOne case.

Fatigue MouseOne	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.5796	1.6179	.01	.2463	.4633	.03
Left Wrist	.8519	2.4221	.01	.5556	1.6186	.01
Left Arm	.1000	.8042	.19	-.1556	.6481	.01

Table A.22: Fatigue statistics for the left arm for the MouseOne condition.

Testing for the null hypothesis yields the table A.23.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.24 is for the MouseTwo case.

Testing for the null hypothesis yields table A.25.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.26 is for the OneTwo case.

Testing for the null hypothesis yields table A.27.

For the left hand, the Midway and End scores are both normally distributed, and yield probabilities less than 0.10. The T-Test for the End score yields 0.01, a strongly significant result. The null hypothesis must therefore be rejected.

The 95% confidence interval for the left hand fatigue rating for the OneTwo condition is (.771..3.562), which barely includes 1 in the range. Taking the mean at face value, it would seem that the increase in fatigue for the left hand under this condition exceeds 1, falsifying hypothesis

Fatigue MouseOne	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Left Hand			.2850	.2850
Left Wrist			.1797	.1797
Left Arm	.847			.6547

Table A.23: Testing null hypothesis for left arm fatigue in the MouseOne condition.

Fatigue MouseTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.0000	.6325	.20	.0000	.6325	.20
Left Wrist	-.1296	.4082	.01	.0000	.6325	.20
Left Arm	-.1296	.4082	.01	-.3148	.5164	.01

Table A.24: Fatigue statistics for the left arm for the MouseTwo condition.

Fatigue MouseTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Left Hand	1.000	1.000		
Left Wrist		1.000	.3173	
Left Arm			.3173	.1797

Table A.25: Testing null hypothesis for left arm fatigue in the MouseTwo condition.

Fatigue OneTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	1.6517	1.9408	.41	2.3423	1.3292	.54
Left Wrist	1.0412	1.6330	.08	1.0741	1.6021	.07
Left Arm	1.5741	2.5820	.01	1.4444	1.9748	.21

Table A.26: Fatigue statistics for the left arm for the OneTwo condition.

Fatigue OneTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Left Hand	.069	.010		
Left Wrist			.1088	.1088
Left Arm		.122	.1797	

Table A.27: Testing null hypothesis for left arm fatigue in the OneTwo condition.

11 for the left hand in this condition. Hypothesis 10 can only be tentatively accepted, because the null hypothesis is accepted by only a slight margin. Testing hypothesis 12 for this case with a paired-samples T-Test, the significance probability of a difference between the two means is 0.530, indicating that the Midway and End means are likely the same.

The tests for the left wrist and the left arm yield probabilities greater than 0.10, so the null hypothesis of means not different from zero should probably be accepted for these body regions.

For hypothesis 13, pairwise comparisons are again made.

Table A.28 presents tests for difference of Midway fatigue scores, with means briefly summarized. The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Fatigue Midway	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Left Hand	.58	.00	1.65	.592	.184	.070T
Left Wrist	.85	-.13	1.04	.093	.591	.045
Left Arm	.10	-.13	1.57	.391	.474	.092

Table A.28: Midway fatigue comparisons between conditions for the left arm.

Table A.29 presents tests for difference of End fatigue scores, with means briefly summarized.

Fatigue End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Left Hand	.25	.00	2.34	.655	.017	.005T
Left Wrist	.56	.00	1.07	.341	.531	.128
Left Arm	-.16	-.32	1.44	.445	.107	.031

Table A.29: End fatigue comparisons between conditions for the left arm.

At the Midway point, there is a statistically significant difference between the MouseTwo case and the OneTwo case for the Left Wrist. At the End point, there are statistically significant differences between OneTwo and the other 2 cases for the Left Hand, and a between the MouseTwo case and the OneTwo case for the Left Arm. The fatigue rating for the Left Wrist is no longer significantly different.

In summary, where the statistical tests indicate a difference between conditions, ranking in order of increasing scores is as follows:

1. MouseTwo
2. MouseOne
3. OneTwo

Hypothesis 13 is false for the Left Wrist at the Midway point, and for the Left Hand and the Left Arm at the End survey. Combined with the previous tests comparing the ratings with 0, there

is good evidence that the Left Hand experiences an increase in fatigue in the OneTwo case. There is moderate evidence that the Left Arm experiences an increase in fatigue in the OneTwo case, because the significance level of the difference from 0 is 0.122 for the Left Arm in the OneTwo case.

A.4 Left Arm Pain

Table A.30 indicates left arm pain ratings for all subjects, with Shapiro-Wilks normality probabilities.

Pain All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.4630	1.4605	.01	.3580	1.2423	.01
Left Wrist	.3457	1.5782	.01	.1728	1.0166	.01
Left Arm	.3642	1.6830	.01	.3210	1.0369	.01

Table A.30: All-subject pain statistics for the left arm.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.31.

Pain All	Midway=0	End=0
Region	2-tail P	2-tail P
Left Hand	.1380	.1380
Left Wrist	.1159	.2249
Left Arm	.1775	.1380

Table A.31: Null hypothesis tests for all-subject left arm pain scores.

All of these means are fairly small, and none is significantly different from 0.

Analyzing by condition yields table A.32 for the MouseOne case.

Pain MouseOne	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	.556	1.619	.01	.611	1.605	.01
Left Wrist	.796	2.442	.01	.611	1.605	.01
Left Arm	.204	.4082	.01	.000	.000	.01

Table A.32: Pain statistics for the left arm for the MouseOne condition.

Since none of these ratings are normally distributed, Wilcoxon tests for the null hypothesis yields table A.33.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.34 is for the MouseTwo case.

Testing for the null hypothesis yields table A.35.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.36 is for the OneTwo case.

Pain MouseOne	Wilcoxon Significance	
Region	Midway=0	End=0
Left Hand	.1797	.1797
Left Wrist	.1797	.1797
Left Arm	.1797	1.0000

Table A.33: Testing null hypothesis for left arm pain in the MouseOne condition.

Pain MouseTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	-.130	.408	.01	-.130	.408	.01
Left Wrist	.000	.636	.20	.000	.632	.20
Left Arm	-.130	.408	.01	.185	.753	.25

Table A.34: Pain statistics for the left arm for the MouseTwo condition.

Pain MouseTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Left Hand			.3173	.3173
Left Wrist	.235	.363		
Left Arm		.175	.3173	

Table A.35: Testing null hypothesis for left arm pain in the MouseTwo condition.

Pain OneTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Left Hand	1.074	1.835	.01	.759	1.329	.01
Left Wrist	.574	1.211	.01	.130	.408	.01
Left Arm	1.519	2.658	.01	.944	1.549	.01

Table A.36: Pain statistics for the left arm for the OneTwo condition.

Since none of these ratings are normally distributed, Wilcoxon tests for the null hypothesis yields table A.37.

Pain OneTwo	Wilcoxon Significance	
Region	Midway=0	End=0
Left Hand	.1797	.1797
Left Wrist	.1797	.3173
Left Arm	.1797	.1797

Table A.37: Testing null hypothesis for left arm pain in the OneTwo condition.

For the left arm as a whole, none of the scores are statistically significantly different from 0, indicating that hypotheses 10, 11, and 12 can be accepted.

For hypothesis 13, pairwise comparisons are again made.

Table A.38 presents tests for difference of Midway pain scores, with means briefly summarized. The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Pain Midway	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Left Hand	.56	-.13	1.07	.093	.924	.093
Left Wrist	.80	.00	.57	.341	1.000	.294
Left Arm	.20	-.13	1.52	.093	.703	.093

Table A.38: Midway pain comparisons between conditions for the left arm.

Table A.39 presents tests for difference of End pain scores, with means briefly summarized.

Pain End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Left Hand	.61	-.13	.80	.093	1.000	.093
Left Wrist	.61	.00	.13	.341	.528	.598
Left Arm	.00	.19	.94	.527	.138	.473

Table A.39: End pain comparisons between conditions for the left arm.

Since there is no statistically significant difference between scores under any condition at both the Midway and End points, Hypothesis 13 holds for left arm pain.

A.5 Right Arm Fatigue

Table A.40 indicates right arm fatigue ratings for all subjects, with Shapiro-Wilks normality probabilities.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.41.

In this case, none of the right arm means differ significantly from zero.

Analyzing by condition yields table A.42 for the MouseOne case.

Fatigue All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.1687	1.2580	.01	.5123	1.1742	.01
Right Wrist	.2000	1.4389	.01	.3580	1.0604	.01
Right Arm	.0926	1.3959	.01	.1296	1.1540	.01

Table A.40: All-subject fatigue statistics for the right arm.

Fatigue All	Midway=0	End=0
Region	2-tail P	2-tail P
Right Hand	.2026	.0929
Right Wrist	.3433	.1614
Right Arm	.7353	.7353

Table A.41: Null hypothesis tests for all-subject right arm fatigue scores.

Fatigue MouseOne	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.5851	1.2166	.16	.4066	1.2497	.46
Right Wrist	.9367	1.5753	.01	.1261	1.6330	.30
Right Arm	.8440	1.7248	.21	.4369	1.4569	.02

Table A.42: Fatigue statistics for the right arm for the MouseOne condition.

Fatigue MouseOne	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Right Hand	.848	.828		
Right Wrist		.924	.5002	
Right Arm	.737			.7150

Table A.43: Testing null hypothesis for right arm fatigue in the MouseOne condition.

Testing for the null hypothesis yields table A.43.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.44 is for the MouseTwo case.

Fatigue MouseTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.1852	.4021	.01	.2222	.4183	.01
Right Wrist	.2963	.8042	.01	.0631	.4916	.01
Right Arm	.0000	.0000	.01	.0000	.0000	.01

Table A.44: Fatigue statistics for the right arm for the MouseTwo condition.

Since none of these ratings are normally distributed, Wilcoxon tests for the null hypothesis yield table A.45.

Fatigue MouseTwo	Wilcoxon Significance	
Region	Midway=0	End=0
Right Hand	.1797	.1797
Right Wrist	.1797	.1088
Right Arm	1.0000	1.0000

Table A.45: Testing null hypothesis for right arm fatigue in the MouseTwo condition.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.46 is for the OneTwo case.

Fatigue OneTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	-.2011	1.8348	.19	.2103	1.4720	.02
Right Wrist	.2593	1.9664	.06	.4444	.8367	.01
Right Arm	.0000	1.8974	.20	.4030	1.5055	.25

Table A.46: Fatigue statistics for the right arm for the OneTwo condition.

Testing for the null hypothesis yields table A.47.

For the right arm as a whole, none of the scores are statistically significantly different from 0, indicating that hypotheses 10, 11, and 12 can be accepted.

For hypothesis 13, pairwise comparisons are again made.

Table A.48 presents tests for difference of Midway fatigue scores, with means briefly summarized. The test results ending in T indicate that the two component means were normally distributed, so a T-Test was used.

Table A.49 presents tests for difference of End fatigue scores, with means briefly summarized.

Since there is no statistically significant difference between scores under any condition at both the Midway and End points, Hypothesis 13 holds for right arm fatigue.

Fatigue OneTwo	T-Test Significance		Wilcoxon Significance	
Region	Midway=0	End=0	Midway=0	End=0
Right Hand	.317			.1088
Right Wrist			.6547	.1797
Right Arm	1.000	.611		

Table A.47: Testing null hypothesis for right arm fatigue in the OneTwo condition.

Fatigue Midway	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Right Hand	.59	.19	-.20	.867	.434T	1.000
Right Wrist	.94	.30	.26	.503	.403	.446
Right Arm	.84	.00	.00	.107	.837T	1.0000

Table A.48: Midway fatigue comparisons between conditions for the right arm.

A.6 Right Arm Pain

Table A.50 indicates right arm pain ratings for all subjects, with Shapiro-Wilks normality probabilities.

Again, none of these means are normally distributed. The null hypothesis tests are in table A.51.

All of these means are fairly small, and none is significantly different from 0.

Analyzing by condition yields table A.52 for the MouseOne case.

Since none of these ratings are normally distributed, Wilcoxon tests for the null hypothesis yield table A.53.

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.54 is for the MouseTwo case.

Since none of these ratings are normally distributed, Wilcoxon tests for the null hypothesis yield table A.55. .

None of these tests yield probabilities less than 0.05, so the null hypothesis of means not different from zero should probably be accepted.

Table A.56 is for the OneTwo case.

Since none of these ratings are normally distributed, Wilcoxon tests for the null hypothesis yield table A.57.

For the arm as a whole, none of the scores are statistically significantly different from 0, indicating

Fatigue End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Right Hand	.41	.22	.21	.929	.303	.324
Right Wrist	.13	.06	.44	.549	.655	.857
Right Arm	.44	.00	.40	.252	.868	.527

Table A.49: End fatigue comparisons between conditions for the right arm.

Pain All	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.3519	1.0911	.01	.1543	.7078	.01
Right Wrist	.1605	.5469	.01	.0247	.3557	.01
Right Arm	.1049	.6853	.01	.0370	.5053	.01

Table A.50: All-subject pain statistics for the right arm.

Pain All	Midway=0	End=0
Region	2-tail P	2-tail P
Right Hand	.0759	.2733
Right Wrist	.0679	.7893
Right Arm	.4631	.8927

Table A.51: Null hypothesis tests for all-subject right arm pain scores.

Pain MouseOne	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.0847	.8042	.05	.3333	.8000	.01
Right Wrist	.0854	.4899	.02	-.0778	.4690	.03
Right Arm	.3921	.4967	.02	.2407	.4320	.01

Table A.52: Pain statistics for the right arm for the MouseOne condition.

Pain MouseOne	Wilcoxon Significance	
Region	Midway=0	End=0
Right Hand	.1088	.1797
Right Wrist	.1088	.6547
Right Arm	.1088	.1797

Table A.53: Testing null hypothesis for right arm pain in the MouseOne condition.

Pain MouseTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.1296	.4082	.01	.0000	.0000	.01
Right Wrist	.0000	.0000	.01	.0000	.0000	.01
Right Arm	-.1296	.4082	.01	-.1296	.4082	.01

Table A.54: Pain statistics for the right arm for the MouseTwo condition.

Pain MouseTwo	Wilcoxon Significance	
Region	Midway=0	End=0
Right Hand	.3173	1.0000
Right Wrist	1.0000	1.0000
Right Arm	.3173	.3173

Table A.55: Testing null hypothesis for right arm pain in the MouseTwo condition.

Pain OneTwo	Midway			End		
Region	M-Est	StDev	S-W	M-Est	StDev	S-W
Right Hand	.0000	1.7512	.08	.1296	.9832	.06
Right Wrist	.2593	.8165	.01	.1296	.4082	.01
Right Arm	.1296	.9832	.06	.0000	.6325	.20

Table A.56: Pain statistics for the right arm for the OneTwo condition.

Pain OneTwo	Wilcoxon Significance	
Region	Midway=0	End=0
Right Hand	.4227	.6547
Right Wrist	.3173	.3173
Right Arm	.6547	1.0000

Table A.57: Testing null hypothesis for right arm pain in the OneTwo condition.

that hypotheses 10, 11, and 12 can be accepted.

For hypothesis 13, pairwise comparisons are again made.

Table A.58 presents tests for difference of Midway pain scores, with means briefly summarized.

Pain Midway	Means			Tests		
Region	M1	M2	I2	M1-M2	M1-I2	M2-I2
Right Hand	.08	.13	.00	.7518	.6074	.8485
Right Wrist	.40	.60	.26	.0578	.3900	.3173
Right Arm	.39	-.13	.13	.0449	.2830	.5982

Table A.58: Midway pain comparisons between conditions for the right arm.

Table A.59 presents tests for difference of End pain scores, with means briefly summarized.

At the Midway point, there is a statistically significant difference in Right Arm scores between the MouseOne and MouseTwo cases. However, the actual difference between the scores is very small, so it is likely that this is a chance occurrence. For the remainder, since there is no statistically significant difference between scores under any condition at both the Midway and End points, Hypothesis 13 holds for right arm pain.

Pain End	Means			Tests		
Region	M1	M2	12	M1-M2	M1-12	M2-12
Right Hand	.33	.00	.13	.1396	.3900	1.0000
Right Wrist	-.08	.00	.13	1.0000	.5283	.3173
Right Arm	.24	.41	.00	.0927	.3900	.5982

Table A.59: End pain comparisons between conditions for the right arm.

Appendix B

Tutorial For Alias and Two-Handed Users

B.1 Preface

Starting with the next section, this appendix quotes the content of the tutorial paper that was given to the subjects that were to use the Mouse-Based interface and the Two-Handed interface (MouseTwo subjects). A similar tutorial is given for the MouseOne subjects. Because the interfaces are different, there is a fairly independent introduction to each.

The OneTwo subjects read a somewhat shorter tutorial that first introduced the One-Handed Interface, then introduced THRED by referring to the One-Handed material. A similar strategy was adopted in Chapter 5 to introduce the One-Handed system in reference to THRED. The OneTwo tutorial is not included here.

B.2 Introduction

The purpose of this document is to introduce the basic free-form surface editing system, and then to explain the two different interfaces to the basic geometry. The interfaces are as follows:

1. Mouse-Based Interface: This interface presents the 3 traditional orthographic views, and a perspective view of the object being modeled. All interaction uses the mouse.

2. Two-handed 3D Trackers: The user is presented with a single perspective view of the object, and the user has a 3-button 3D tracker in each hand. The user's dominant hand selects and manipulates the object, and the non-dominant hand does menu selection and reorients the model with the second tracker.

In the experiment, you will perform some simple editing tasks using each interface. You will perform the simple task many times using one interface, then the same task using the second interface, and again using the first interface. The purpose of repeating the task is to help us understand how well a user can learn each interface while safely disregarding the effects of learning the task. By the time you complete the task on the first interface, you should be quite practiced at the task, and the differences in times should be due entirely to the ease-of-use of the interface.

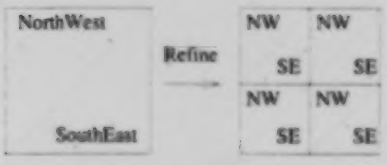


Figure B.1: A single quad and its refinement into four sub-quads.

B.3 Geometry

These design systems all use a common surface geometry system. At its simplest, the surface is simply a rectangular grid of vertices, with each vertex connected to its North, South, East and West neighbors. This rectangular edge connection scheme provides a collection of quadrilaterals (quads) that can be reshaped by moving one or more of their respective vertices. To generate flat polygons, each quad is subdivided into a NorthWest and a SouthEast triangle, allowing a direct edge connection from a vertex to its NorthEast and SouthWest neighbors.

Each quad can be subdivided into exactly 4 sub-quads, resulting in 8 corresponding triangles. Each of these can be further refined into four sub-sub-quads, and so on. As a result, some basic root-level quads can contain many child vertices.

There are four basic operations which can be performed on each vertex or quad:

The *refine* operation creates four child quads for a quad, the *move* operation changes the vertex location, the *erase* operation removes a quad from the mesh, and the *add row/column* operation adds a row or a column of quads to the periphery of the root-level mesh.

To create a surface model, the user starts with a single quad at the origin and commences adding rows and columns, refining quads, and moving vertices. The user builds the desired surface by iteratively modifying, refining, and examining the surface given the available interface tools.

B.3.1 Surface Display

In order to help the user visually locate the vertices, the NorthWest triangle of a quad is drawn a darker shade than the SouthEast triangle. Each level has its own unique hue, so for example the root level is red, level 1 is yellow, level 2 is green, level 3 is blue, and so on.

Two types of textures are mapped onto the surface of all triangles, a random texture and a contour texture. The contour texture produces a series of equal-height contour lines on the surface, which helps the user gauge the height of features and to get a feel for the relative sizes of features. This texture presents a ruler pattern of meters and tenths of meters on the surface. There is a wide tick mark at zero meters, a medium tick mark at 0.5 meters, and light tick marks at the other 1/10 meter locations.

On selected quads, you can also display a ruler along the edges of each triangle. This ruler will allow you to visually measure edge length. There is also an angle measure available, which will allow you to measure the three angles of each selected triangle. You can display none, either, or both of these measurements on any quad by selecting the desired quads and enabling ruler display or angle display. You may then freely select other quads for other operations, since the rulers and/or angle

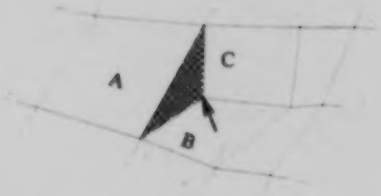


Figure B.2: A crack (the shaded triangle) appears between squares A, B and C because the marked vertex joining B and C should be on A's edge. Therefore, these vertices cannot be moved.

measures stay active on a quad until disabled.

B.3.2 Selection

To reshape the surface, the user must first select which vertices or quads are to be changed. You can select one vertex at a time, or you can select a rectangular region of vertices. This rectangular region is based on mesh connectivity, not on the current shape of the surface, so the rectangular region may look quite irregular. To select a rectangular region, you press a button on the vertex at one corner, and you sweep over to the other corner and release the button. The rectangle may be degenerate: a single row or column of points is also valid. Depending on the interface, the button is on either the mouse or the 3D tracker.

The set of currently selected vertices remain selected until you pick nothing. You can also unselect a vertex by picking it. That is, you can add a vertex to the selected set by picking it, and you can remove it by picking it again. Similarly, any selected vertices within a newly-picked rectangular region will be removed from the currently selected set. You will be shown how this works.

The set of currently-selected vertices is outlined by yellow piping along its periphery. All vertices within this yellow region or on the boundary may be moved. Quads which have all four of their vertices in the selected set may be refined or deleted. Single vertices cannot be deleted, nor can they be refined.

Red piping indicates that selected vertex cannot be moved because moving it would create a hole in the surface. These vertices occur on the border with a square of shallower refinement, as shown in figure B.2. These vertices can be used to surround a quad that is to be refined, however.

B.3.3 Reshaping

When you press button 2, the squares of the selected area can be reshaped according to the current reshaping operator. Currently the four reshaping operations are *move everything in tandem*, *cone* reshape, *scale* reshape, and *adopt parent*. Again, button two is either the middle mouse button, or the middle 3D tracker button.

The tandem move operation is useful for grabbing a block of squares and moving them all at once. Snap gridding is also available on a 1/10 meter resolution. The nearest vertex that is within the snap threshold is snapped to the 1/10 meter line. This occurs in all 3 dimensions.

When moving perpendicular to the plane, the cone reshape operator acts as if a conical tent

were being erected centered at the middle vertex.

The scale operation scales the distance of the non-center vertices from the center vertex. This operator can be used to exaggerate or minimize the spikiness of the selected features, and in the second experimental task, you will use this operator frequently.

The adopt parent operation causes the selected squares to adopt the geometry of their parent squares. The selected squares realign into groups of four on their parent squares as if the parents had just been refined. The purpose of this operator is to reduce the amount of high-frequency surface detail, and is used as a "start over" operator. This is not under continuous control, and takes place as soon as the user presses button 2.

B.3.4 Reshaping with Orientation

When you press button 3 on the 3D tracker in your dominant hand, only the tandem reshape mode is active, and you can directly manipulate both the position and orientation of the selected vertices in tandem. This operation is available through the *rotate* menu item in the mouse-based interface.

B.3.5 Constrained Reshaping

In the 3D tracker interfaces, reshaping operators can move in all 3 dimensions at once, or can also be constrained to move along one of the 3 axis lines or in one of the three axis planes. In the mouse-based system, 2D or 1D reshaping can be selected by choosing the appropriate orthographic view.

At this point, the interfaces differ enough from each other that it is best to present each interface and to explain all its operations.

B.4 Mouse-Based Interface

The Mouse-Based Interface presents the three traditional Top, Right, and Front orthographic views of the object, and a perspective view. The user may view all four windows at once, or may pick one view to occupy the whole screen. Each window has a little title bar with some icons that can be used to control the view of the scene. In the orthographic windows, the operators are Zoom, Translate and Fullscreen. In the perspective window, there is also a tumble operator that allows you to rotate the scene, as shown in figure B.3. To activate one of these operators, press the left mouse button while the mouse cursor is in the icon. The cursor will then disappear, and while you drag the mouse, the scene will move. When you release the mouse button, the cursor will reappear over the icon. Clicking on the Fullscreen icon will make the window full screen, and clicking it again will return to the 4-window view.

At the bottom of the screen is a menu bar that contains popup menus for all of the operations. Most menus will remember what you last selected, and this is noted in the small rectangle below each menu button. Some menu operations are also *sticky*, in the sense that when you select the operation, it stays active until you select something else. The main sticky operations are **Pick Vertex** and **reshape**.

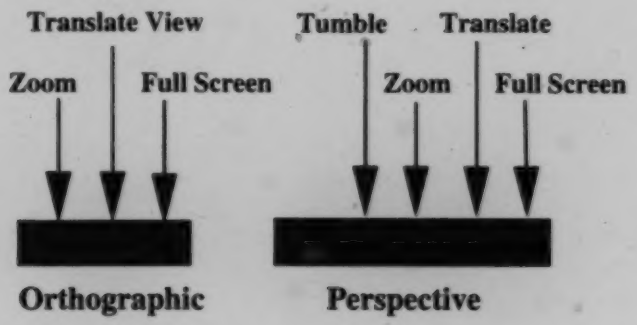


Figure B.3: View control icons in the window title bars.

The *File* menu allows you to save your model, retrieve it, create a new model and quit.

The *Reshape Operators* menu allows you to select which of the reshape operators will be applied when you move multiple vertices. The current operator is printed in the rectangle below the button.

The *Pick* menu has either **Pick Vertex** or **Pick nothing**. The Vertex operator stays active until something else requiring mouse activity in one of the 4 views is activated. Typically, Pick Vertex stays active until one of the *Xform* menu items is selected. Picking is done by putting the mouse near one vertex, pressing the left mouse button, dragging the cursor to the opposite corner of the selection rectangle, and releasing the mouse button. The cursor does not have to be exactly positioned on the vertex – the nearest vertex is picked. Single mouse clicks on a vertex add and remove the vertex from the selected set. **Pick nothing** causes the selected set to be cleared.

The *Xform* menu contains two items: *reshape* and *rotate*. Reshape applies the reshape operator to the selected set of vertices. In orthographic views, dragging with the left mouse button pressed moves the vertices in tandem with the cursor as you would expect. The middle mouse button allows just horizontal movement, and the right button allows just vertical movement. In the perspective view, the left button translates in X, the middle does Y translation and the right does Z translation.

Rotate rotates the selected set about one of the three axes: Left rotates about X, middle rotates about Y, and right rotates about the Z axis. The rotation axis is actually centered at the centroid of all the selected vertices. Both reshape and rotate are sticky, and stay active until some other sticky operation is selected.

The *Modify Squares* menu contains three instantaneous commands: *Refine*, *Add Y Row*, and *Add X Row*. Refine creates 4 sub quads for each quad that has all four vertices in the selected set. Add X and Add Y split the first row or column of root-level squares into two rows or columns. This is the only way to add squares to the mesh.

The *Annotation Tools* menu allows you to control the appearance of the model, and toggles snap gridding. *Tgl grid* turns on or off the display of the regular grid in the orthographic windows. *Tgl snap* turns on or off the 1/10 meter snap grid. *Arc On* and *Arc Off* turn on or off the arc display on each quad that has all four vertices in the selected set. *Ruler On* and *Ruler Off* does the same for rulers. *Tgl random* turns on or off the display of the random surface texture, and *Tgl contour* controls the display of the contour texture. These last two are global effects, and apply to all quads

at once.

The *Delete* menu allows you to immediately delete all quads that have all four vertices in the selected set.

The *Views* menu allows you to adjust the view of any of the four windows on the screen. *Reset View* resets the view of a window to the startup view, and *Previous* sets the view to what it was before the last view-changing operation. For both of these, click on the window that you want to change after selecting the operation from the menu. *Track* and *Zoom* work in the orthographic windows in a manner similar to the title bar icons. *Dolly* works in an orthographic window by allowing you to sweep out a rectangular region with the left mouse button that will fully occupy the window, effectively zooming in. Sweeping out a rectangle with the right mouse button has the inverse effect of zooming out. The *Tumble* operator works in the perspective window similar to the way the title bar tumble works. The remaining operators of *Yaw/pitch*, *Azimuth/elevation*, and *Twist* are only used in the perspective view.

The *Window* menu allows you to select which window to display full-screen, or you can select *All* to display all four.

The *Level Display* window allows you to display all levels of refinement, or to restrict display and vertex selection up to and including the selected level. When the user has restricted vertex selection to a maximum level of refinement, children of the maximum quads are drawn using the maximum level color, but the geometry of the descendant vertices. That is, when the level is restricted, just the color changes to reflect the restriction and to delineate the selectable points.

B.5 Two-Handed 3D Tracker Editor

When two-handed editor starts up, the user is presented with a window containing two sub-windows and the top menu bar. There is only one pull-down menu in the menu bar, containing traditional file operations such as load, save, and quit, and the New surface command.

Below the menu bar is the refinement level window, which displays all of the levels of refinement that have been made to the surface, and shows which levels are currently selectable. The refinement window is only 100 pixels tall, and is used for quick reference and for quick interactions to change the current level of refinement. This window also shows the current reshape operator on the left.

The majority of the screen space is taken up by the main window, which displays the work area and the left and right cursors.

In the two-handed tracker system, you hold one tracker in your right (dominant)[1] hand, and use it as a 3D tool or probe to select and manipulate vertices. The left hand holds a second tracker.

The way to think about this is that the left *Bat*[2] provides *context* of various kinds, and the right bat does *picking and fine manipulation*. Each bat is represented in the virtual space by a 3D cursor that moves within the 3D space of the object being designed. The left bat's cursor is visually distinguished from the right cursor in that the right cursor has a probe attached to it, and the left

[1] For the sake of rhetorical convenience, I'll assume you are right handed, and will call the right hand dominant, and the left hand non-dominant. This is reversed for left-handed people.

[2] A Bat is a flying mouse.

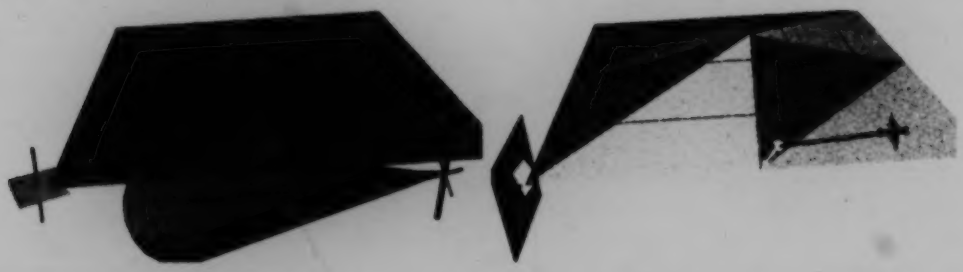


Figure B.4: Left: Line constraint mode is active. Selected vertices can only move vertically. The right cursor shows the selection cone with its central probe axis, and an arrow pointing from the cursor to the nearest vertex.
Right: Plane constraint mode is active. Selected vertices may move in the horizontal plane shown by the left cursor. The probe on the right cursor intersects the surface, and the hot spot is indicated by a small arrow from the intersection point.

does not. There is some potential for confusion between which bat is which, so there is a small patch of velcro on the left bat, and the right bat has no velcro.

B.5.1 The Right Bat

The right bat operations are

- Control point selection and selection adjustment.
- Reshaping the selected area.
- Reshaping the selected area with orientation.

Control point selection is accomplished using the *probe*, which is attached to the right cursor. The probe is represented by a narrow cylindrical shaft, and the user controls the position and orientation of the probe with the right bat. The vertex that is closest to the probe axis is the current *hot spot*, and this vertex is highlighted by drawing an arrow from the intersection point on the surface to the hot spot. This highlight arrow lies in the surface of the triangle where the intersection occurs. The right image of figure B.4 shows the probe and the hot spot arrow.

If no objects intersect the probe axis, the probe visually switches to object selection mode. To select objects, we draw the probe axis to some arbitrary length, and we draw a translucent cone which has a radial spread angle of 10 degrees. The probe is the cone axis, as shown in the left image of figure B.4. We attach to the cursor a spotlight that has identical visual geometry to the translucent cone. As the user sweeps the probe about the scene, the objects that fall within the cone intersect visually with the cone wall, and are highlighted by the spotlight. Since we are interested in selecting vertices, we draw an arrow from the right cursor to the nearest control point. When there is an intersection between the probe axis and a surface, we turn off the translucent cone and the spotlight, because both tend to obscure surface detail.

Button 1 (the *nose* button) on the right bat allows you to select rectangular collections of vertices. Press the right nose button when the hot spot is at one corner, and drag the probe to the opposite corner and release the right nose button. Double-click on the right nose button to select nothing.

If you find it hard to twist your right had around to reach a desired vertex, you can customize the way that the probe attaches to the cursor by pressing the "a" key (for Align probe). This instantly causes the probe to point directly into the screen, so the way to maximize your comfort is to hold the right bat in a comfortable position in your hand, and hit the "a" key.

Similarly, hitting the "o" key instantly moves both cursors to a central "sweet spot" in the editing space, so you can put your hands at a comfortable position and jump the cursors to the sweet spot by hitting the "o" key (for re-Origin). In fact, it moves both cursors by the same amount towards the center of the sweet spot, so that the separation between the two still corresponds to the separation of your hands. Hold both bats close together to move both cursors into the sweet spot.

Pressing right button 2 (the middle or *dorsal* button) grabs the selected vertices and uses the right bat to move them around in 3D. The currently-selected reshape operator is used to move the vertices. This movement can be constrained to move along a line or within a plane.

Pressing right button 3 (the *tail* button) grabs the selected vertices and uses the right bat to translate and rotate them in 3D. Only the tandem reshape operator allows rotation. The translation component can be constrained to move along a line or within a plane.

B.5.2 The Left Bat

The left bat does the following:

- Set the current constraint mode.
- Menu selection
- Manipulate the position and orientation of the entire scene.

Constrained reshaping is controlled by first selecting which one of three modes is desired: *no* constraints, *line* constraints, and *plane* constraints. The left hand provides both the type of constraint and the geometry of the constraint. Clicking the left nose button moves from the current constraint mode to next in a three-step cycle.

Initially, the right cursor can manipulate vertices with all 3 positional degrees of freedom. When the left nose button is clicked, the system moves into line-constraint mode, in which right-handed reshaping operations are constrained along one of the three major axes. Both cursors change their shape to reflect line constraint mode, with the left cursor changing to a short pole through a square, and the right cursor adding a short pole that snaps to the current constraint axis (figure B.4). The left cursor does not snap but remains tied to the left bat, because the canonical axis closest to the bat's Z vector (the pole axis in the left cursor) determines the constraint axis.

Thus while the left bat is geometrically stating the constraint axis, the right axis can be making a selection and starting the reshape operation. In order to avoid any inadvertent changes in snap axis, the constraint axis remains fixed while the reshape operation takes place, so the left hand does not need to hold a steady orientation during reshaping.

Similarly, in plane constraint mode the canonical plane of motion is stated continuously and directly by the orientation of the left hand, and remains constant while the right hand is doing a reshape. In this case, the right cursor shows a small square that is snapped to the constraint plane, and the left cursor shows a much larger snapped plane as shown in figure B.4. The left cursor also has a small square rotating freely with the left bat, to help you get to the desired orientation.

B.5.3 Viewing The Surface

To examine the surface as you edit it, press and release the left tail button. This will attach the surface to the left cursor, and will allow you to turn it around and look at it from all sides. Pressing and releasing the left tail button will detach the surface from the left hand. This operation corresponds to the view control menu in the Mouse-Based Interface.

B.5.4 The Sundial Menu

Pressing the left dorsal button pops up a pie-shaped *sundial* menu. This menu allows you to select the reshape operator, to perform instant operations like refinement and deletion, and to control the display of the surface. All of the menu operations are instantaneous, or are mode changes.

The sundial menu (figure B.5) is a popup menu controlled by the orientation of the left bat. The menu choices are arrayed on the circular plate of the sundial, each on its own pie-shaped sector. The desired item is picked by pivoting the *shadow stick* about its base so that the stick's endpoint lies visually in front of the sector. The base of the shadow stick is located at the center of the plate, and when the menu first pops up, the stick is aligned with the line of sight, pointing directly at the user. Around the base of the stick is an inner circle which the user can point the stick at to indicate *no selection*. The position of the sundial is fixed rigidly to the bat, and the dial lies parallel to the screen.

Hierarchical menus are supported using the sundial menu. As figure B.5 shows, a child menu is represented by an outward-pointing arrowhead.

The user moves the shadow stick into the parent sector to pop up the child. As a visual reminder of where the parent is located, the child pops up with its center located over the arrowhead of the parent sector. The child is popped up only if the rotational velocity of the bat drops below a certain threshold while in the parent sector. This ensures that the shadow stick is not in motion when the child appears. Therefore, you must stop rotating the left bat to allow a child menu to pop up. You can select child items in the same way as you would select from the main menu – rotate the left bat to move the shadow stick into the appropriate sector, and release the left dorsal button when that sector's label highlights. To select nothing, rotate the stick to the center region and release the left dorsal button.

B.5.5 Menu Items

The items on the main menu are the 5 reshape operators, and two submenus. The *Annotations* submenu has the same operations as does the Mouse-Based Interface *Annotation Tools* menu:

Figure B.5: The sundial menu. Tandem (box) reshaping is currently selected. Clockwise from tandem is cone, scale, scale to median plane, adopt parent, blank, the Annotations submenu and the Misc submenu.

Snap Tgl turns on or off the 1/10 meter snap grid. *Arc On* and *Arc Off* turn on or off the arc display on each quad that has all four vertices in the selected set. *Ruler On* and *Ruler Off* does the same for rulers. *All Off* turns off both ruler and arc display. *Random Tex* turns on or off the display of the random surface texture, and *Contour Tex* controls the display of the contour texture. These last two are global effects, and apply to all quads at once.

The *Misc* menu contains 4 instantaneous editing commands, and 4 viewing control commands. The editing commands are *Refine* and *Erase*, which operate on each quad that has all four vertices in the selected set. *Divide X* and *Divide Y* split the first row or column of root-level squares into two rows or columns.

The 4 viewing commands are *Scale Up*, *Scale Down*, *Spot Align*, and *Re-Origin*. Scale Up doubles the displayed size of the surface, and Scale Down shrinks it by half. To align the probe so that it is directed into the screen, you can select Spot Align, or press the "a" key. The Spotlight Alignment operation occurs when you release the left dorsal button, so you should adopt a relaxed pose with your right hand before you release the button.

Re-Origin instantly moves both cursors to the central sweet spot. In fact, it moves both cursors by the same amount towards the center of the sweet spot, so that the separation between the two still corresponds to the separation of your hands. Hold both bats close together to move both cursors into the sweet spot. This can also be accomplished by pressing the "o" key.

B.5.6 Level Window

You can select what level of refinement will be displayed or selectable by moving the left bat above the top edge of the main window into the refinement level window. The cursor becomes a simple one-dimensional slider represented by a white arrow. Press the left dorsal button to select a

Figure B.6: Top: The button-enhanced Bat. Bottom: Four postures for holding the bat. The X axis points up and the Z axis points to the left in the bottom left three pictures.

refinement level. Sweep the right bat back and forth while the left dorsal button is pressed to select all levels.

B.5.7 Hints

Some manual skill is needed to reorient the bat appropriately, especially in using the Sundial menu. The best hint we can give is *relax*: Before each operation, such as selecting an item from the menu, or orienting the constraint direction, relax your hand so it is in a comfortable orientation *before you start*. That way, you won't have to turn the bat very far to get to the desired orientation. If you do not do this, you will find that your hand will get further and further out of your comfort zone, and you will find the bat hard to use.

Second, *use fingertips*. Considerable control is available from reorienting the bat using just your fingertips, and it will reduce the amount of effort you have to expend to move the bat.

Appendix C

Survey Sheets

C.1 Survey Sheets

This appendix shows the survey pages that were used condict the various steps of the user survey. The originals were designed to fill the pages with very narrow margins, so these pages were scaled down by a factor of 0.75 to conform with thesis format and margin guidlines. These pages should be scaled up by a factor of 1.33 to regain the original size. To verify that the correct size has been reached, each rating scale should be 10cm wide.

The survey questions given here are all the unique pages that exist in the survey. To greate a complete survey booklet, 3 copies of the pain and fatigue survey pages would appear, as outlined in chapter 5. The survey pages that follow are:

1. Demographic and Edinburgh Handedness survey.
2. Experience survey.
3. Pain and Fatigue survey for torso. This page appears 3 times in the booklet.
4. Pain and Fatigue survey for left and right arms. This page appears 3 times in the booklet.
5. Midway general familiarity and preferences. The AutoCAD familiarity question occurs only on this page. The presentation order is Mouse first, Two-Handed second.
6. Midway detailed preferences. Mouse is assigned to the negative end of the scale, Two-Handed to the positive end.
7. End general familiarity and preferences. The AutoCAD familiarity question does not appear. The presentation order is Two-Handed first, Mouse second.
8. End detailed preferences. Two-Handed is assigned to the negative end of the scale, Mouse to the positive end.

This booklet would be used for a subject that first used the Mouse-Based system followed by the Two-Handed system in the first block of trials. The pages in the complete booklet would be 1, 2, 3, 4, 5, 6, 3, 4, 7, 8, 3, 4.

For the reverse presentation order, "Two-Handed" and "Mouse-Based" would be switched throughout the booklet. The detailed preference pages differ only in the interfaces that are assigned to scale endpoints, so they can be used at both the Midway or End surveys depending on presentation order.

Handedness Inventory

Student ID _____ **Birth Date** _____ **Sex** ____
Program _____

Please indicate your preferences in the use of hands in the following activities *by putting + in the appropriate column.* Where the preference is so strong that you would never try to use the other hand unless absolutely forced to, *put* ++. If in any case you are really indifferent, *put + in both columns.*

Some of the activities require both hands. In these cases the part of the task, or object, for which hand is wanted is indicated in brackets.

Please try to answer all the questions, and only leave a blank if you have no experience at all of the object or task.

		LEFT	RIGHT
1	Writing		
2	Drawing		
3	Throwing		
4	Scissors		
5	Toothbrush		
6	Knife (without fork)		
7	Spoon		
8	Broom (upper hand)		
9	Striking Match (match)		
10	Opening Box (lid)		
i	Which foot do you prefer to kick with?		
ii	Which eye do you use when using only one?		

L Q	
Decile	

Experience Questionnaire

Please estimate the amount of experience you have with the following types of computer systems and skills as accurately as possible. Answer the *"When did you start using.."* questions with the month and year that you first started doing the given task at the given frequency, or answer *"Never"*.

Answer the *"How many days/month/years..."* questions with an estimate of the amount of time you spent regularly using the given system. Use the most sensible time unit, and do *not* count time periods where you used the given system less than once per week.

For example, weekly use of AutoCAD in May–August 1994 and May–August 1995 counts as 8 months of experience, not 1.6 years.

School experience counts, but casual or random use does not.

1: When did you start using a computer on a weekly or more frequent basis? _____

2: When did you start using a mouse on a weekly or more frequent basis? _____

3: When did you start using AutoCAD on a monthly or more frequent basis? _____

4: How many days/months/years have you used AutoCAD? _____

5: If you have used other 3D design programs, when did you start? _____

What programs ? _____

How many days/months/years have you used these programs? _____

6: *Counting only the last 5 years* – How many days/months/years of free–hand drawing experience do you have? _____

7: *Counting only the last 5 years* – How many days/months/years of drafting experience do you have? _____

8: How many days/months/years experience do you have with using a 3D tracker system such as a Polhemus or Ascension tracker, or a Logitech 3D mouse ? _____

9: How many days/months/years have you programmed or designed 3D computer graphics? _____

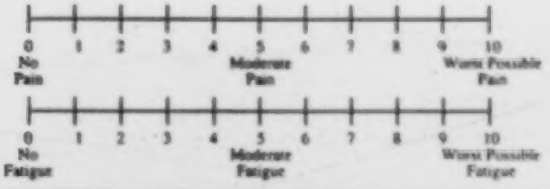
Subjective Ratings

This sheet asks you to rate the level of physical pain and fatigue that you are experiencing right now. You will be presented with this questionnaire more than once.

Mark your ratings by drawing a pen stroke through the rating scale, or by circling the desired tickmark on scale.

Rate your levels of Pain and Fatigue:

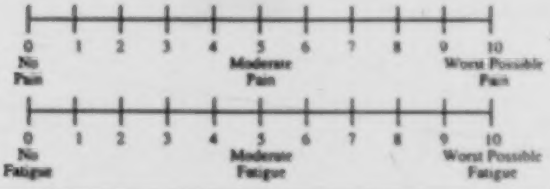
1: Eyes and face


0 1 2 3 4 5 6 7 8 9 10
No Pain
Moderate Pain
Worst Possible Pain
0 1 2 3 4 5 6 7 8 9 10
No Fatigue
Moderate Fatigue
Worst Possible Fatigue

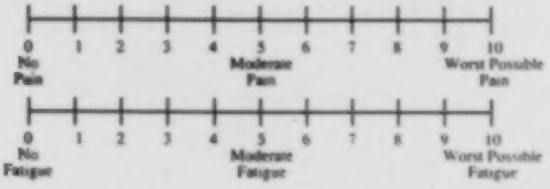
2: Neck

0 1 2 3 4 5 6 7 8 9 10
No Pain
Moderate Pain
Worst Possible Pain
0 1 2 3 4 5 6 7 8 9 10
No Fatigue
Moderate Fatigue
Worst Possible Fatigue

3: Shoulders

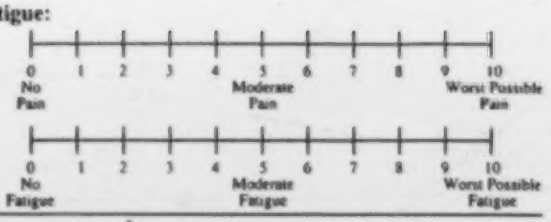

0 1 2 3 4 5 6 7 8 9 10
No Pain
Moderate Pain
Worst Possible Pain
0 1 2 3 4 5 6 7 8 9 10
No Fatigue
Moderate Fatigue
Worst Possible Fatigue

4: Back

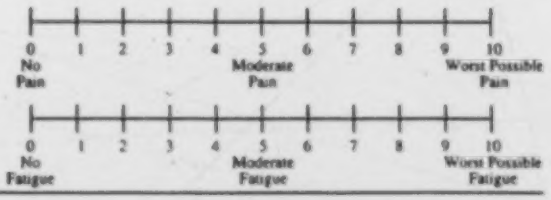

0 1 2 3 4 5 6 7 8 9 10
No Pain
Moderate Pain
Worst Possible Pain
0 1 2 3 4 5 6 7 8 9 10
No Fatigue
Moderate Fatigue
Worst Possible Fatigue

Rate your levels of Pain and Fatigue:

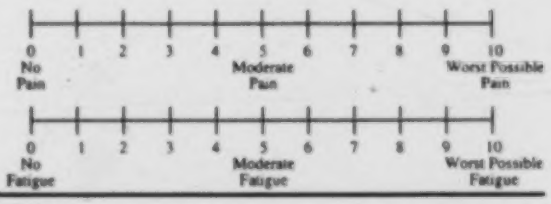
5: Left hand


0 No Pain
1 2 3 4 5 Moderate Pain 6 7 8 9 10 Worst Possible Pain
0 No Fatigue
1 2 3 4 5 Moderate Fatigue 6 7 8 9 10 Worst Possible Fatigue

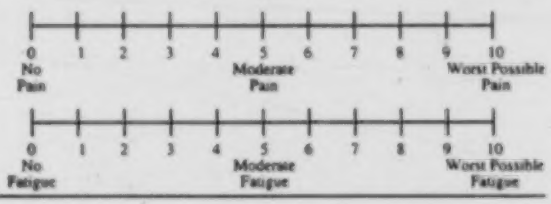
6: Left wrist


0 No Pain
1 2 3 4 5 Moderate Pain 6 7 8 9 10 Worst Possible Pain
0 No Fatigue
1 2 3 4 5 Moderate Fatigue 6 7 8 9 10 Worst Possible Fatigue

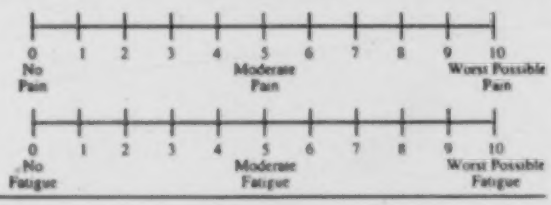
7: Left arm


0 No Pain
1 2 3 4 5 Moderate Pain 6 7 8 9 10 Worst Possible Pain
0 No Fatigue
1 2 3 4 5 Moderate Fatigue 6 7 8 9 10 Worst Possible Fatigue

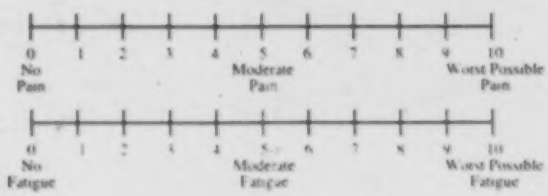
8: Right hand


0 No Pain
1 2 3 4 5 Moderate Pain 6 7 8 9 10 Worst Possible Pain
0 No Fatigue
1 2 3 4 5 Moderate Fatigue 6 7 8 9 10 Worst Possible Fatigue

9: Right wrist

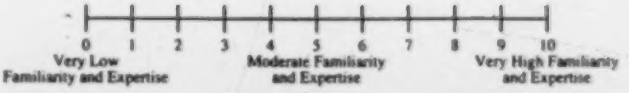

0 No Pain
1 2 3 4 5 Moderate Pain 6 7 8 9 10 Worst Possible Pain
0 No Fatigue
1 2 3 4 5 Moderate Fatigue 6 7 8 9 10 Worst Possible Fatigue

10: Right arm

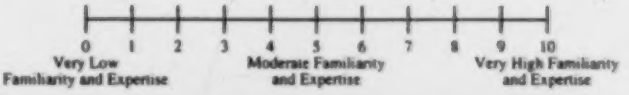


Rate your familiarity and expertise:

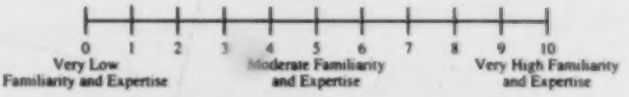
1: AutoCAD:


0 1 2 3 4 5 6 7 8 9 10
Very Low Familiarity and Expertise
Moderate Familiarity and Expertise
Very High Familiarity and Expertise

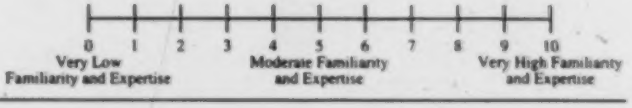
2: Mouse–Based Interface:


0 1 2 3 4 5 6 7 8 9 10
Very Low Familiarity and Expertise
Moderate Familiarity and Expertise
Very High Familiarity and Expertise

3: Two–Handed 3D Interface:

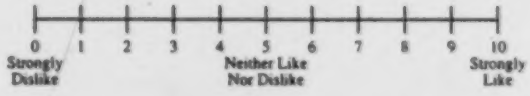

0 1 2 3 4 5 6 7 8 9 10
Very Low Familiarity and Expertise
Moderate Familiarity and Expertise
Very High Familiarity and Expertise

4: Creating Regular 3 by 3 grid:


0 1 2 3 4 5 6 7 8 9 10
Very Low Familiarity and Expertise
Moderate Familiarity and Expertise
Very High Familiarity and Expertise

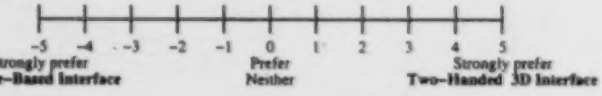
Rate your preferences:

1: Mouse–Based Interface:

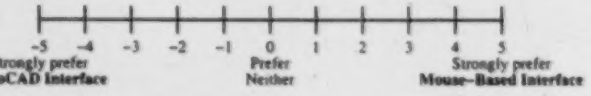


2: Two–Handed 3D Interface:

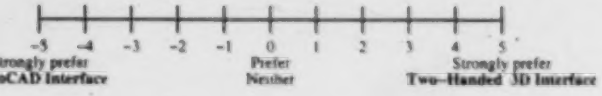
3: Compare the experimental systems to each other:



4: Compare the **AutoCAD interface** to the **Mouse–Based Interface**:

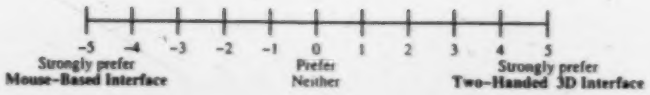


5: Compare the **AutoCAD interface** to the **Two–Handed 3D Interface**:

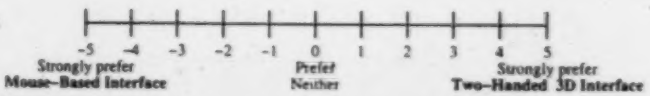


Rate your preferences:

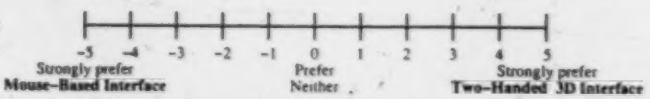
6: Vertex Selection:



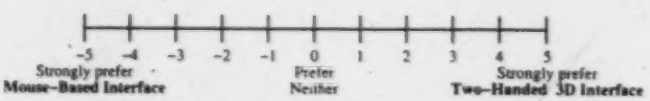
7: Moving and reshaping vertices **without constraints**



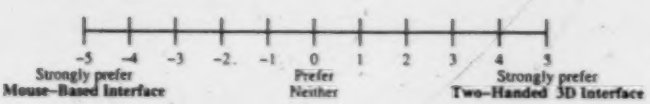
8: Moving and reshaping vertices **with constraints**



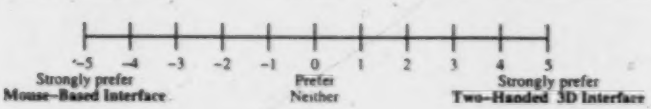
9: Activating rulers



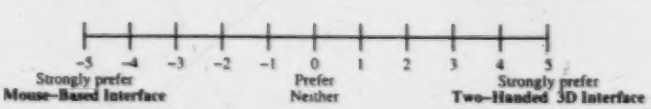
10: View control:



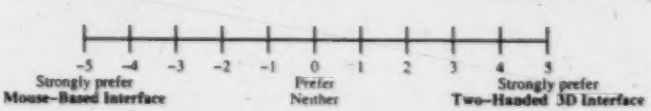
11: Examining surface



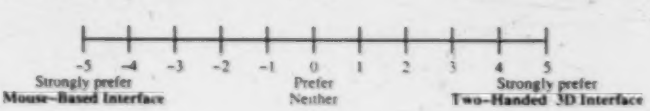
12: Examining rulers



13: Menu appearance:

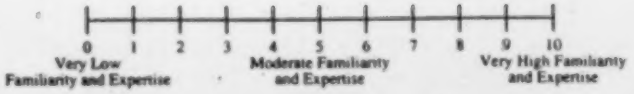


14: Menu selection:

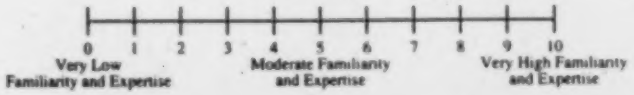


Rate your familiarity and expertise:

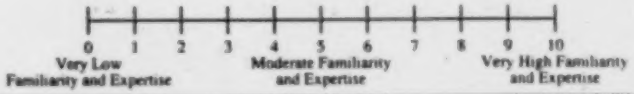
1: Two–Handed 3D Interface:


0 1 2 3 4 5 6 7 8 9 10
Very Low Familiarity and Expertise
Moderate Familiarity and Expertise
Very High Familiarity and Expertise

2: Mouse–Based Interface:

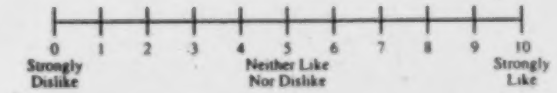

0 1 2 3 4 5 6 7 8 9 10
Very Low Familiarity and Expertise
Moderate Familiarity and Expertise
Very High Familiarity and Expertise

3: Creating Regular 3 by 3 grid:


0 1 2 3 4 5 6 7 8 9 10
Very Low Familiarity and Expertise
Moderate Familiarity and Expertise
Very High Familiarity and Expertise

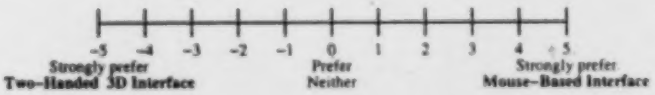
Rate your preferences:

1: Two–Handed 3D Interface:

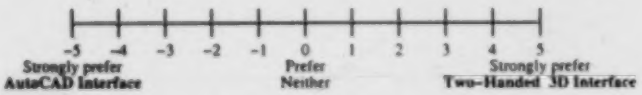


2: Mouse–Based Interface:

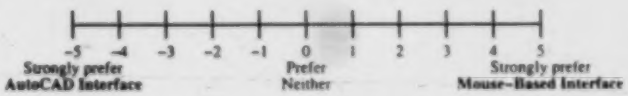
3: Compare the experimental systems to each other:



4: Compare the **AutoCAD interface** to the **Two–Handed 3D Interface**:

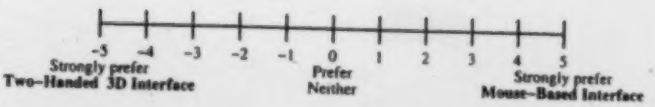


5: Compare the **AutoCAD interface** to the **Mouse–Based Interface**:

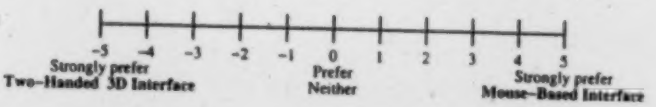


Rate your preferences:

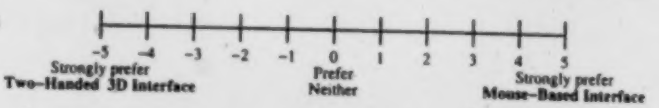
6: Vertex Selection:



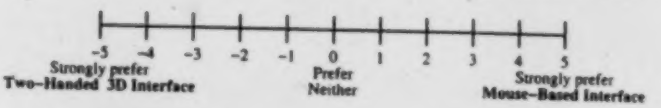
7: Moving and reshaping vertices **without constraints**



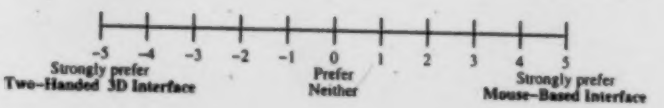
8: Moving and reshaping vertices **with constraints**



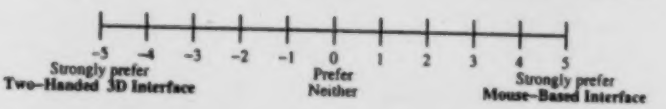
9: Activating rulers



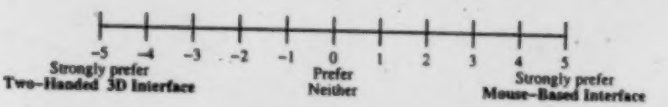
10: View control:



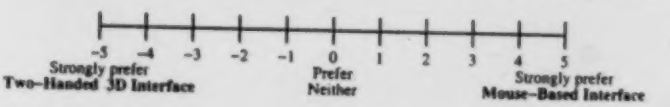
11: Examining surface



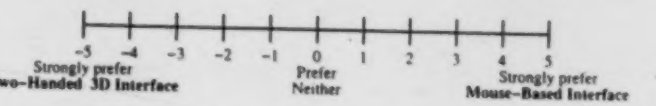
12: Examining rulers



13: Menu appearance:



14: Menu selection:



Index

3-Draw: A Tool for Designing 3D Shapes, 6, 16, 41, 52
3D Design Reference Framework, 9
3DM: A Three Dimensional Modeler Using a Head-Mounted Display, 13

A Hand Gesture Interface Device, 13
A Study in Interactive 3-D Rotation Using 2-D Control Devices, 11
A Study in Two-Handed Input, 16
A Taxonomy of See-Through Tools, 17
Alias Research Inc., 3, 4, 8, 18, 55, 132
An Empirical Comparison of Pie vs. Linear Menus, 46, 49
An Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus, 51
ARCBALL: A User Interface for Specifying Three-Dimensional Orientation Using a Mouse, 11
Art of Natural Graphic Man-Machine Conversation, The, 7
Arthur, Kevin, 33
Artificial Reality with Force-feedback: Development of Desktop Virtual Space with Compact Master Manipulator, 15
Assessment and Analysis of Handedness: The Edinburgh Inventory, The, 68, 77
Asymmetric Division of Labor in Human Skilled Bimanual Action: The Kinematic Chain as a Model., 2

Barr, Alan, 23
Barr, Alan H., 19
Bartels, Richard, 18, 23, 28, 29, 136
Baudel, Thomas, 17
Bechmann, D., 22
Bechmann, Dominique, 22, 23
Bier, Eric A., 17
Bier, Eric Allan, 10, 12, 40
Black, Alison, 12
Blanchard, Chuck, 13
Booth, Kellogg S., 33
Borrel, P., 22
Borrel, Paul, 23
Braunstein, M.L., 33
Bryson, Steve, 13
Butterworth, J., 13
Buxton, William, 16, 17, 39, 50, 51, 133
Buxton, William A S, 51

CAGD's top ten: What to watch, 18
Callahan, Jack, 46, 49
Card, Stuart K., 28, 61, 121
Castellan Jr., John, 80, 138
Chan, P., 7
Chen, Michael, 11
Chua, Beng Choon, 6, 14
Clark, James H., 13
Conner, D. Brookshire, 9, 10, 19
contributors, 17, 9
Conway, Matthew J., 41
Coquillart, Sabine, 21

Davidson, A., 13
Decoupled Simulation in Virtual Reality with The MR Toolkit, 26
Deering, Michael, 15, 33, 49
Deformation of *n*-dimensional objects, 22

Deformation of Solids with Trivariate B-Splines, 20
Designing Surfaces in 3-D, 13
Direct manipulation of free-form deformations, 21
Direct Manipulation Techniques for Objects Using 2D Locator Devices, 9

Elastically Deformable Models, 23
Empirical Comparison of Pie vs. Linear Menus, An, 46, 49
Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus, An, 51
Evaluating Stereo and Motion Cues for Visualizing Information Nets in Three Dimensions, 133
Evans, Kenneth B., 11
Extended Free-Form Deformation: A Sculpturing Tool for 3D Geometric Modeling, 21

Farin, Gerald, 18
Feiner, Steven K, 7
Fishkin, Ken, 17
Fishtank Virtual Reality, 33
Fitts, P. M., 8, 9, 43, 49
Fleischer, Kurt, 23, 24
Foley, J. D., 7
Foley, James. D., 7
Foley, James D, 7
Forsey, David, 18, 23, 28, 29, 136
Fowler, Barry, 19
Franck, Glenn, 133
Free-Form Deformation of Solid Geometric Models, 20

Geometric manipulation of tensor product surfaces, 19
Geometric Modeling Using Six Degrees of Freedom Input Devices, 3, 4, 6, 11, 16, 37, 42, 46, 132
Global and Local Deformations of Solid Primitives, 19
Goble, John C., 6, 41
Green, Mark, 3, 4, 6, 11, 16, 26, 37, 39, 42, 46, 132
Griessmair, Josef, 20
Guiard, Yves, 2

Hand Gesture Interface Device, A, 13
Harvill, Young, 13
Hench, S., 13
Herndon, Kenneth P, 9, 10, 19
Hierarchical B-Spline Refinement, 18, 23, 28, 29, 136
High Resolution Virtual Reality, 15, 33
Hinckley, Ken, 6, 41
HoloSketch VR Sketching System, The, 49
Hopkins, Don, 46, 49
Hsu, William M., 21
Hughes, John F, 7
Hughes, John F., 21
Human Factors of Computer Graphics Interaction Techniques, The, 7
Human Performance Using Computer Input Devices in the Preferred and Non-Preferred Hands, 17

Influence of Muscle Groups on Performance of Multiple Degree-of-Freedom Input, The, 133
Influential Companions. Effects of one strategy on another in the within-subjects designs of cognitive psychology, 66, 123
Information Capacity of the Human Motor System in Controlling the Amplitude of Movement, The, 8, 9, 43, 49
Interaction Techniques for a Virtual Workspace, 6, 14
Interactive Design of 3D Models with Geometric Constraints, 10

Interactive Shadows, 9
Iwata, Hiroo, 15

JDCAD: A Highly Interactive 3D Modeling System, 3, 4, 6, 16, 37, 39, 42, 46, 132
Jensen, T. W., 18
Jessome, Danny R., 1, 26, 113
Jolley, Janina, 65
Joy, K. I., 21

Kabbash, Paul, 17, 39, 133
Kassel, Neal F., 6, 41
Kaufman, Henry, 21
Kling, J.W., 9
Kurtenbach, Gordon, 50, 51
Kurtenbach, Gordon P, 51

Lanier, Jaron, 13
Layer Tool: Support for Progressive Design, 12
Liang, Jiandong, 3, 4, 6, 11, 16, 26, 37, 39, 42, 46, 132
Logitech Inc., 28

MacKenzie, I. Scott, 17
Milgram, Paul, 39, 133
Mitchell, Mark, 65
Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture, 23, 24
Moran, Thomas P., 28, 61, 121
Mountford, S. Joy, 11
Myers, Brad A., 16

Newell, Allen, 28, 61, 121
Ng, Hern, 6, 14
Nielson, Gregory M., 9, 18
Norusis, Marija J., 78

Olano, T.M., 13
Oldfield, R C, 68, 77
Olsen, Dan R., 9

Parry, Scott R., 20
Passive Real-World Interface Props for Neurosurgical Visualization, 6, 41
Pausch, Randy, 6, 41
Petersen, C. S., 18
Physically Based Models with Rigid and Deformable Components, 23
Physically-Based Models with Rigid and Deformable Components, 23
Platt, John, 23
Poston, Timothy, 6, 14
Poulton, E C, 66, 123
Practical curves and surfaces for a geometric modeler, 18
"Prince" Technique: Fitts' Law and Selection Using Area Cursors, The, 39, 133
Prototyping for Tiny Fingers, 12
Purgathofer, Werner, 20

Quadtree and Related Hierarchical Data Structures, The, 29

Rappoport, Ari, 23
Regularization of Inverse Visual Problems Involving Discontinuities, 24
Rettig, M, 12
Riggs, Lorrin A., 9
Robbins, Daniel C, 9, 10, 19
Roberts, Andrew, 6, 16, 41, 52

Sachs, Emanuel, 6, 16, 41, 52
Samet, H., 29
Schmandt, Chris, 14
Sederberg, Thomas W., 20
Sellen, Abigail, 11
Sellen, Abigail J, 17, 51
Sellers, Don, 100, 132
Serra, Luis, 6, 14
Shaughnessy, John J, 65, 66
Shaw, Chris, 6, 26
Shneiderman, Ben, 46, 49
Shoemake, Ken, 11

Siegel, Sidney, 80, 138
"Silk Cursor": Investigating Transparency for 3D Target Acquisition, 39
Simple Constrained Deformations for Geometric Modeling and Interactive Design, 23
Sittas, E, 9
Skitters and Jacks: Interactive 3D Positioning Tools, 10, 12, 40
Snibbe, Scott S, 9, 10, 19
Space Deformation Models Survey, 22, 23
Spatial Input/Output Correspondence in a Stereoscopic Computer Graphics Work Station, 14
Stoakley, Richard, 41
Stone, Maureen C., 17
Stoops, David, 6, 16, 41, 52
Study in Interactive 3-D Rotation Using 2-D Control Devices, A, 11
Study in Two-Handed Input, A, 16
Sun, Yunqi, 26

Tablet-based Valuators that Provide One, Two, or Three Degrees of Freedom, 11
Tanner, Peter P., 11
Taxonomy of See-Through Tools, A, 17
Terzopoulos, Demetri, 23, 24
The "Prince" Technique: Fitts' Law and Selection Using Area Cursors, 39, 133
The "Silk Cursor": Investigating Transparency for 3D Target Acquisition, 39
The Art of Natural Graphic Man-Machine Conversation, 7
The Assessment and Analysis of Handedness: The Edinburgh Inventory, 68, 77
The HoloSketch VR Sketching System, 49
The Human Factors of Computer Graphics Interaction Techniques, 7
The Influence of Muscle Groups on Performance of Multiple Degree-of-Freedom Input, 133
The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement, 8, 9, 43, 49
The Quadtree and Related Hierarchical Data Structures, 29
The Virtual Workbench: Dextrous VR, 14
Two-Handed Input in a Compound Task, 17
Two-Handed Input in a Compound Task, 17
Two-Handed Polygonal Surface Design, 6

User Learning and Performance with Marking Menus, 50, 51
Using Deformations to Explore 3D Widget Design, 10, 19
Using the Bat: A Six Dimensional Mouse for Object Placement, 1, 26, 113
Utilizing parametric hyperpatch methods for modeling and display of free-form solids, 21

van Dam, Andries, 7, 9, 10, 19
van Emmerik, Maarten J. G. M., 10
Virtual Reality on a WIM: Interactive Worlds in Miniature, 41
Virtual Workbench: Dextrous VR, The, 14
Visible Planning on Paper and on Screen: The Impact of Working Medium on Decision-Making by Novice Graphic Designers, 12

Wallace, V. L., 7
Wallace, Victor L., 7
Ware, Colin, 1, 26, 33, 113, 133
Waterworth, John A., 6, 14
Watkins, M. A., 18
Wein, Marceli, 11
Weiser, Mark, 46, 49
Witkin, Andrew, 23
Wong, Y. Y., 12

Zechmeister, Eugene B, 65, 66
Zeleznik, Robert C, 9

Zhai, Shumin, 39, 133
Zimmerman, Thomas G., 13